NIST-GCR-94-643

MATHEMATICAL MODELING OF HUMAN EGRESS FROM FIRES IN RESIDENTIAL BUILDINGS

Michael M. Kostreva
Clemson University
Clemson, SC 29631

Issued June 1994
January 1994

Sponsored by:
U.S. Department of Commerce
Ronald H. Brown, *Secretary*
Technology Administration
Mary L. Good, *Under Secretary for Technology*
National Institute of Standards and Technology
Arati Prabhakar, *Director*

Notice

This report was prepared for the Building and Fire Research Laboratory of the National Institute of Standards and Technology under grant number 60NANBoD1023. The statements and conclusions contained in this report are those of the authors and do not necessarily reflect the views of the National Institute of Standards and Technology or the Building and Fire Research Laboratory.

FINAL TECHNICAL REPORT

"MATHEMATICAL MODELING OF HUMAN EGRESS FROM FIRES IN RESIDENTIAL BUILDINGS"

Grant Number 60NANB0D1023

Date: January 15, 1994

Report Prepared by: Dr. Michael M. Kostreva

NIST Scientific Officer: Mr. Rick Peacock

Sponsor: Building and Fire Research Laboratory
National Institute of Standards and Technology

1. Introduction

Over the past several years, the Building and Fire Research Laboratory at the National Institute of Standards and Technology (formerly the National Bureau of Standards), has been developing mathematical models for predicting the environmental conditions which occur as a fire develops and spreads. These models form a significant part of the computer program entitled HAZARD I, which has been made available to the scientific community. Another very significant part of HAZARD I methodology is the modeling of human egress. Human egress modeling has been implemented in HAZARD I by means of simulation of actual human movement, invoking psychological theories and heuristic methodologies to create paths which the humans in the residential building under consideration would likely have taken. As a matter for research, it was proposed to consider mathematical models which would find optimal and/or Pareto optimal (nondominated) paths for human egress in the same situations. These paths may serve as a valuable reference for the designers of buildings. They have many other functions as well. For example, in the case of buildings which may have electronic emergency systems capable of adapting to hazardous conditions and giving the appropriate optimal egress paths to the occupants of the building, these paths may serve as the basis for the information delivered by the system. Finally, optimal paths serve as a standard against which the heuristically generated paths of HAZARD I may be evaluated and validated.

The research supported by this grant has been rather successful at finding mathematical models, appropriate solution techniques, coding the techniques, and making sample applications. In seven different research papers the results of the research have been documented. Six of these papers have appeared in print, while the seventh is submitted for publication. The full copies of the papers are included as appendices of this report, while summaries are presented here for the reader's convenience.

2. Summaries of research papers

[1] "Optimization Models in Fire Egress Analysis for Residential Buildings," in Fire Safety Science - Proceedings of Third International Symposium, (G. Cox and B. Langford, Eds.), Elsevier Applied Science, London and New York, 1991, pp. 805-814, (with M. M. Wiecek and T. Getachew).

Fire hazard analysis models are developed for the study of optimal egress of individual occupants of a residential building which is involved in a fire. Several examples of increasing complexity and realism demonstrate the methodology, which is based on dynamic programming. Finally, there is a multi-attribute risk analysis technique which may be used to evaluate a proposed building design.

[2] "Multicriteria Decision Making in Fire Egress Analysis," Preprints of IFAC/IFORS Workshop on Support Systems for Decision and Negotiation Processes, (R. Kulikowski, Z. Nahorski, J. Owsinski, and A. Straszak, eds.) Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland, June 1992, pp. 285-290 (with M. Wiecek).

A multicriteria dynamic approach to fire egress analysis is presented in an algorithm to generate all nondominated egress paths for all occupants in a building. An application related to a fire in a North Carolina food processing plant is given.

[3] "An Algorithm for Multiple-Objective Path Optimization with Time Dependent Links," Proceedings of 10th International Conference on Multiple Criteria Decision Making, Volume 3, pp. 319- 330, Taipei, Taiwan, July 1992 (with T. Getachew).

A new algorithm for computing all nondominated paths in a network with time dependent vector cost functions is presented. It is an algorithm capable of handling non monotone increasing cost functions, which may also be discontinuous. These advances allow for people-fire interactions such as opening and closing doors and windows. The paper does not contain proofs, due to limitations on space.

[4] "Approximation in Time Dependent Multiple Objective Path Planning," Proceedings of 1992 IEEE International Conference on Systems, Man and Cybernetics, Volume 1, pp. 861-866, Chicago, Illinois, October 1992 (with M. Wiecek).

Here the question of whether a simplified algorithm, based only on constant cost multiple objective dynamic programming subproblems, can be used successfully to approximate the time dependent problems. The results show that it is possible to get the entire nondominated set in some problems with the simplified approach. On the other hand, it was also shown that, in some cases, certain solutions will be missed, while in other cases, dominated points will be found.

[5] "Transient Behavior in Multiple Criteria Path Planning Problems," Proceedings of 1992 IEEE International Conference on Systems, Man and Cybernetics, Volume 1, pp. 867-873, Chicago, Illinois, October 1992 (with T. Getachew).

In this paper the algorithm of [3] is applied to study the behavior of the set of all nondominated paths, as the egress starting time is varied for the individual occupants. This problem is of interest in fire egress analysis, since not all occupants will be alerted at the same time. Certain patterns can be observed in the nondominated paths set, but no final conclusion was drawn.

[6] "Time Dependency in Multiple Objective Dynamic Programming," Journal of Mathematical Analysis and Applications 173, 289-307 (1993) (with M. Wiecek).

This paper contains the algorithms, proofs, and other supporting mathematics for the applications described in [1] and [2]. It is of interest to those who write computer programs implementing the research results, as well as those interested in extending the mathematics to more complex cases.

The paper was accepted without revision for this very prestigious journal.

[7] "Pareto Optimization in Dynamic Networks with Bounded Cost Functions," submitted for publication (with T. Getachew).

The algorithms, proofs, and other supporting mathematics for the results described in [3] and [5] are contained in this paper. It seems that the case of bounded cost functions is very interesting for people-fire interaction modeling. This is because monotonic increasing cost functions, which are handled by an algorithm of reference [6], do not always apply to strong people-fire interactions. For example, if a door is closed in a fire, the monotonic development of the fire may be interrupted. If fire fighting techniques are applied to a fire, similar things happen. Thus it is of interest to understand this more general type of cost function. Using the methods developed in this paper, one may begin to comprehend the most realistic people-fire interactions.

3. Conclusion

The research performed has advanced the state of the art in fire egress analysis , which forms an important part of fire hazard analysis and modeling as embodied in HAZARD I. In doing so, some new mathematical ideas, algorithms and theories have also been advanced. It is particularly satisfying to have new mathematics derived in support of such an important national priority as fire safety, and, as the mathematical knowledge is general, it may be of service to others in the future.

Appendix 1

"Optimization Models in Fire Egress Analysis for Residential Buildings," in Fire Safety Science - Proceedings of Third International Symposium, (G. Cox and B. Langford, Eds.), Elsevier Applied Science, London and New York, 1991, pp. 805-814, (with M. M. Wiecek and T. Getachew).

Optimization Models in Fire Egress Analysis for Residential Buildings

MICHAEL M. KOSTREVA, MALGORZATA M. WIECEK and
TEODROS GETACHEW
Department of Mathematical Sciences
Clemson University
Clemson, South Carolina 29634-1907, USA

ABSTRACT

Fire hazard analysis often includes the use of mathematical models of the egress of the individual occupants of a structure which is involved in a fire. In this paper, we introduce some mathematical optimization models which produce as output at least one path for each occupant which is optimal with respect to some measurement. Network based models of increasing levels of complexity and realism are demonstrated by means of a sequence of examples. These examples include single attribute constant costs, single attribute time-varying costs, two attribute constant costs, and finally two attribute time-varying costs imposed on network links. Dynamic programming functional equations which are based on the principle of optimality are presented. A multi-attribute analysis is proposed to evaluate a building design with respect to evacuation paths.

KEY WORDS: Egress models, dynamic programming, networks, multiple-objective optimization.

INTRODUCTION

The mathematical modeling of egress from a building on fire falls into one of two general categories. The first is primarily descriptive, focusing on the progress of the occupants over a given time span. The second category has as its focus the determination of globally optimal trajectories of egress. Its results, useful in establishing benchmarks for actual evacuation, must draw heavily upon the theory of optimization.

Fairly extensive work has been done on the simulation of egress from a building on fire. All the models that we have looked at have included one or more of the following components. i) A network description of the building, ii) a set of heuristics for determining evacuee decisions, iii) a quantity that the evacuees are interested in minimizing, iv) input from a separate simulation that keeps track of the progress of the fire and v) an algorithm. Berlin, Dutt and Gupta [1] simulate emergency evacuation under different combinations of rescue and egress options. The required input is a set of descriptors for travel distances between various nodes in the building. As output this model gives measures of escape route congestion, the proportion of safe residents within a given time interval and the estimated evacuation time. Stahl [2] considers the effects of resident ambulatory capacities on safe egress. In EXITT, [3], [4], occupant capacities are expanded to include age, sex, and sleep; the building description has provisions for smoke detectors and background noise in addition to node and

FIRE SAFETY SCIENCE—PROCEEDINGS OF THE THIRD INTERNATIONAL SYMPOSIUM, pp. 805–814

exit information. Occupants, having chosen a course of action, move from one node to the next according to a shortest-distance algorithm. The output is a description of the decisions and movements of occupants over time. HAZARD I, [5], [6], incorporates EXITT to simulate occupant decisions and actions, FAST [7] to perform fire and smoke calculations, and TENAB, to calculate the impact of toxins on the occupants.

Relatively little work has been done in the determination of globally optimal egress paths. EVACNET+ [8] is a computer program that determines time-optimal evacuation plans of a building. Taking building, occupant, node, travel time as input, it provides statistical output for quantities such as total evacuation time, number of successful evacuees, periods of building evacuation etc. It does not however trace the movement of individuals; neither does it consider people-fire interactions. The purpose of this paper is to address these two important issues.

Optimal egress involves multiple criteria decisions in a dynamic environment. Kostreva and Wiecek [9] propose that methods of multiple objective optimization and dynamic progamming may be applied to this problem. In this paper, we demonstrate applications of methods in optimal routing [10], and shortest-path algorithms in dynamic network [11] and multi-objective settings.

The body of this paper is organized into three sections. The main tool in the first section is The Principle of Optimality [12]: "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision". First it is shown how this principle was adapted to networks, via an iterative scheme that successively approximates the optimal paths from all nodes to a destination node. It is then described how to handle the time-dependent case. Finally, this is generalized to solve the multi-objective problem, where it is of interest to optimize more than one quantity. In the Examples section, optimal egress paths in static and dynamic cost networks are obtained for a simple ranch house. Using this same building, we determine non-dominated ('best') egress paths in the two-attribute case. The subsequent section applies a multi-attribute graphical analysis to locate high and low risk zones of the ranch house.

MATHEMATICAL FOUNDATIONS

The mathematical models we consider are simplified versions of a residential building which use a superimposed network to represent locations within and around the building. The network consists of a set of nodes $\{1,2,...,N\}$ and a set of links which indicate connections between nodes $\{(i_1, i_2),(i_3, i_4),... \}$. The links have a direction, indicated by the order of the indices. That is, (3,4) is the link from node 3 to node 4, while (4,3) is the link from node 4 to node 3. A path from node i_0 to node i_p is a sequence of arcs $P=\{ (i_0, i_1), (i_1, i_2), ... ,(i_{p-1}, i_p) \}$ in which the initial node of each arc is the same as the terminal arc of the preceding arc in the sequence and $i_0 , ... , i_p$ are all distinct nodes. A path represents a connected curve along the floor of the building.

Each link carries one or more attributes (i.e. time to travel, distance to travel, etc.) which we think of as costs. The costs apply to all occupants. Each occupant in the building will be initially located at a unique node of the network, denoted an origin node. An egress path is a set of links from the origin node i_1 of the occupant to a destination node N, which is outside the building. That is, the egress path has the form

$$\{(i_1, i_2),(i_2, i_3),(i_3, i_4),...,(i_r, N)\}.$$

The mathematical optimization models we use are designed to find, for each occupant, the egress path (not necessarily unique) which has the minimum cost, or if there are two attributes, a path which is not dominated by any other available egress path. One path P is dominated by another, say P', if P has the same or higher values in each attribute than the path P'. If there are several non-dominated paths for a given occupant, it is of interest to know at least one of them.

Now a dynamic programming approach to finding optimal egress paths in a network will be described, first for the simplest case and then for the more complex models. Dynamic programming, as we will apply it, is based on the principle of optimality: A path which is optimal contains only optimal links. From an intuitive point of view, a path will be proposed initially, and this path will be improved by discarding links which are not optimal and choosing links which are optimal among the choices available.

Let the integer N denote the number of the destination node. We wish to compute the minimum cost path from each node i to N. The costs to travel from node i to node j along link (i,j) are given and are denoted t_{ij} . If f_i is the optimal total cost to go from node i to node N, then the principle of optimality [10] requires that

$$f_i = \min_{\substack{j \neq i \\ (i,j) \text{ a link}}} \{ f_j + t_{ij} \} \quad \text{for } i=1,..,N-1,$$

$$f_N = 0.$$

One approach is to use successive approximations on the above system of nonlinear equations to compute an optimal path. This set of equations may be solved as follows:

Let $f_i^{(0)} = \max_{j \neq i} t_{ij}$ for i=1,..,N-1 and $f_N^{(0)} = 0$.

Then, for k =1,2,3,...

$$\text{let } f_i^{(k)} = \min_{\substack{j \neq i \\ (i,j) \text{ a link}}} \{ f_j^{(k-1)} + t_{ij} \} \quad \text{for } i=1,..,N-1,$$

$$\text{and} \quad f_N^{(k)} = 0.$$

Now consider the case of time dependent costs t_{ij} . A type of function which is of major interest in egress modeling is a step function. If one is coordinating an egress model with a fire physics model, such as in HAZARD I [6], the time history of smoke developing in the building is available. For example, if at time t*, node j* becomes impassable due to intense smoke, then we may model this situation as the step function

$$t_{ij*}(t) = \begin{cases} t_{ij*}, \text{ if } t < t^* \\ M, \text{ if } t \geq t^*, \text{ and M is a large positive number.} \end{cases}$$

For this type of analysis to apply one must be able to determine, for each optimal path $\{(i, i_1),..., (j^*, i_s),...\}$ the node j* arrival time T_{ij*} of the occupant with origin node i. There are two steps:

1) Optimize the network to compute all optimal paths from all nodes i to node N, ignoring the step function, keeping the constant t_{ij} . Let all paths which pass through j*, have arrival times T_{ij*} which are less than t*. In this case, the problem is solved and no path needs to be recomputed.

2) For each link entering the impassable node j*after t*, substitute M for the cost. For other links entering j*, and for links leaving j*, leave the costs unmodified. Optimize again. The result will be new optimal paths which avoid the impassable node, if any exist. If there are none, costs will include terms in M.

The final type of dynamic programming we shall consider is multiple objective dynamic programming. Two or more attributes may be associated with each link of the network we are analyzing: time to travel, distance of travel, concentration of toxic chemical on the link, density of smoke, average temperature on the link, etc. In our model we seek, for each origin node, paths to the destination node which are non-dominated. In terms of the principle of optimality, non-dominated paths contain only non-dominated links. Generally, there will be a set of non-dominated paths for us to compute. Parallel to the treatment above for the single attribute case, an equation arises to describe the principle of optimality:

$$\{\underline{F}_i\} = \text{vmin}_{\substack{j \neq i \\ (i,j) \text{ a link}}} \{ \{\underline{F}_j\} + \underline{t}_{ij} \} \quad \text{for } i=1,..,\ N-1,$$

$$\{\underline{F}_N\} = \{ \underline{0} \}.$$

In the above equation, $\{\underline{F}_i\}$ stands for the set of non-dominated paths from node i to node N, $\underline{t}_{ij}$ is the vector of attributes associated with the link from node i to node j and the symbol 'vmin' stands for vector minimization, or finding the non-dominated paths. As in the earlier example, successive approximation may be used to solve these equations. Since the steps are so similar to the scalar case, they will not be repeated here. The method of dynamic programming for a similar multiple attribute network problem is discussed in [13].

The most complex model we consider is a multiple attribute network with time dependent attributes. This is the most relevant to fire egress analysis, yet it has not appeared in the literature in a useful form. Again the most relevant type of time dependence seems to be a step function, which can indicate an impassable node. The solution is by dynamic programming once again; however, it combines two of the methodologies described earlier in this section: time-dependent analysis and searching for non-dominated paths. The technical details of the extended form of dynamic programming may be found in [9]. An example to illustrate the application is contained in the next section.

EXAMPLES

The optimization models presented in the previous section are now applied to analyze the fire egress in a simple ranch house. Figure 1 depicts the floor plan of the house and nodes that have been assigned to rooms or secondary locations within rooms. It is assumed that occupants can be located at the nodes 1,2, . . . , 6, and can leave the house either through the door in the dining area, door in the living area or window in the bedroom. Thus, an egress path could be any path leading from node i, (i=1,2,...,6), to the destination node N=7 located outside the house.

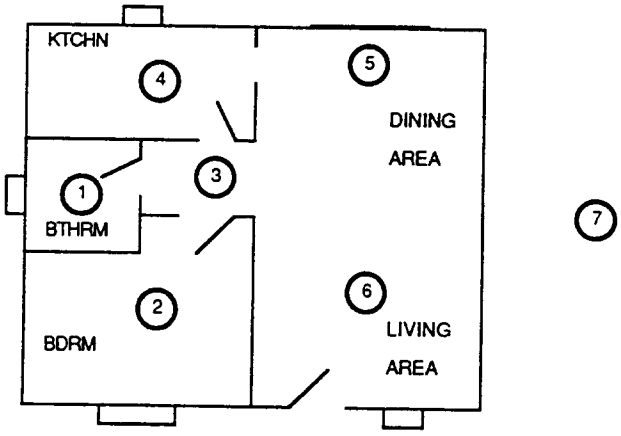


FIGURE 1. Floor plan of a ranch house with arbitrary assignment of nodes.

Figure 2 shows the egress network associated with the house. The number and direction of links results from assumed evacuation routes and directions between the rooms.

For example, nodes 2 and 3 are connected by two links (2,3) and (3,2) since one can move back and forth between the bedroom and the hallway, whereas only one link connects nodes 1 and 3 since leaving the bathroom is the only movement of interest between these nodes.

Below we present four cases that illustrate the modeling concepts and show different optimal evacuation paths.

Case 1: Single Attribute Network with Constant Costs.

Figure 2 also identifies the travel cost t_{ij} on each link (i,j) in the network. One can interpret this cost as the (constant) amount of time necessary to travel from node i to node j. Applying the dynamic programming approach, we can find an optimal (fastest) path from each origin node to the destination node. These optimal paths are depicted in Figure 2 and have thickened links.

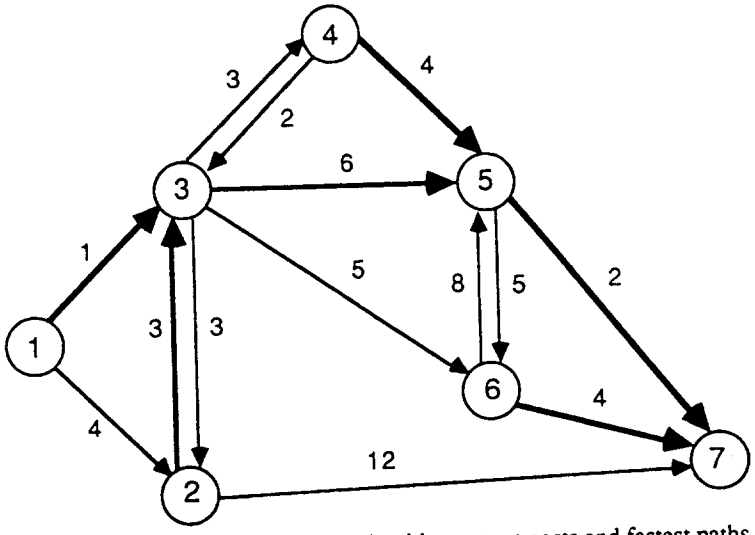


FIGURE 2. Single attribute network with constant costs and fastest paths.

The fastest paths for each origin node i=1,2,...,6 are respectively as follows:
{(1,3), (3,5), (5,7)},
{(2,3), (3,5), (5,7)},
{(3,5), (5,7)},
{(4,5), (5,7)},
{(5,7)},
{(6,7)}.

Consider now the network given in Figure 2 with another attribute which we can think of as (constant) distance between the nodes. Figure 4 shows the network with two attributes, time and distance, assigned to each link. Note that distances (second attribute) on each link in any pair of bilateral links are equal whereas their respective times (first attribute) do not need to be equal. Now we can find optimal (shortest) path in the network with respect to the second attribute only. They are:

{(1,2), (2,7)},
{(2,7)},

{(3,2), (2,7)},
{(4,5),(5,7)},
{(5,7)},
{(6,7)}.
Observe that the shortest paths are not identical with the fastest paths.

Case 2: Single Attribute Network with Time Dependent Costs.

Consider again the network given in Figure 2 and assume that node 5 is impassable. According to the analysis in the previous section one should determine the node 5 arrival time of occupant travelling from some origin node. 7, 9, 6, 4 are the arrival times at node 5 while travelling along the fastest paths from nodes 1, 2, 3, 4 respectively. If node 5 becomes impassable at time t*=4, then we solve a new fastest path problem with modified costs substituted for the costs on links (3,5), (4,5) and (6,5). Figure 3 depicts the original network with node 5 impassable and new optimal paths.

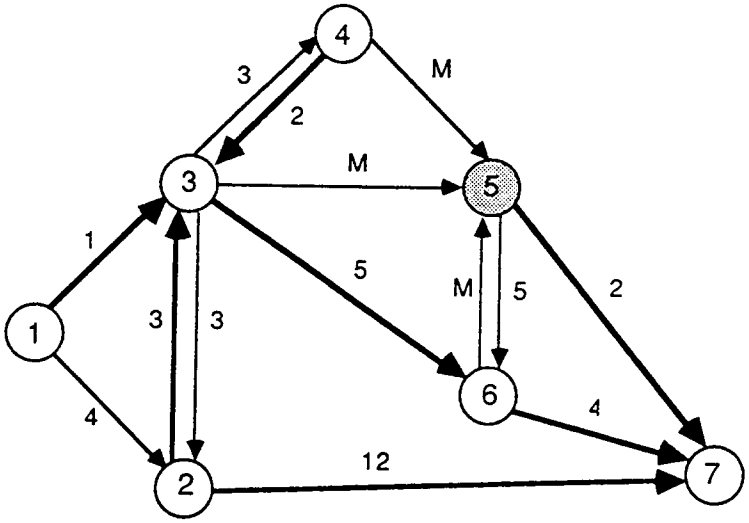


FIGURE 3. Single-attribute network with time dependent costs and new fastest paths.

The new optimal paths that avoid node 5 are:
{(1,3), (3,6), (6,7)},
{(2,3), (3,6), (6,7)} and {(2,7)},
{(3,6), (6,7)},
{(4,3), (3,6), (6,7)},
{(5,7)},
{(6,7)}.

We assume that links (5,6) and (5,7) are available only to occupants originally located at node 5. Hence the path {(5,7)} is also optimal. Note that links (4,3) and (3,6) became optimal as a result of the new structure of the network and that there are two optimal (equally fast) paths from node 2 to node 7 avoiding node 5.

Case 3: Two-Attribute Network with Constant Costs.

Figure 4 illustrates also the multiple objective model discussed in the previous section if both attributes (time and distance) are associated simultaneously with each link of the network. In this case we look for non-dominated paths for each origin node that we can generate by applying vector dynamic programming. There are 11 non-dominated paths for all the origin nodes. We list them below for each origin node i:

{(1,2), (2,7)}, {(1,3), (3,5), (5,7)}, {(1,3), (3,4), (4,5), (5,7)},
{(2,7)}, {(2,3), (3,5), (5,7)},
{(3,2), (2,7)}, {(3,4), (4,5), (5,7)}, {(3,5), (5,7)},
{(4,5), (5,7)},
{(5,7)},
{(6,7)}.

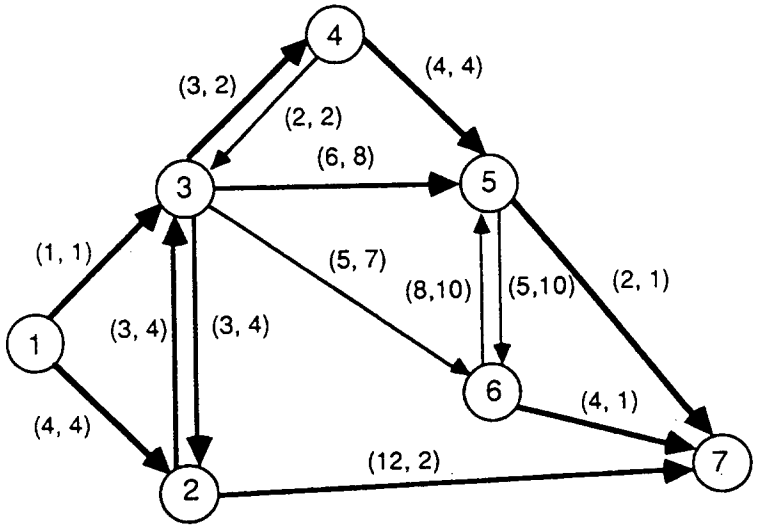


FIGURE 4. Two-attribute network with constant costs and non-dominated paths.

Notice that for nodes 1, 2, and 3 additional paths have been discovered by the multi-attribute network model. These were not found in the single attribute model solution.

Case 4: Two-Attribute Network with Time Dependent Costs.

The final case considered includes a two-attribute network and a step function imposed on node 5 with t*=4. Figure 5 depicts the network with impassable node 5 and adjusted costs on links leading to this node.

The vector dynamic programming approach generates 9 non-dominated paths as follows:

{(1,2), (2,7)}, {(1,3), (3,6), (6,7)},
{(2,7)},
{(3,2), (2,7)}, {(3,6), (6,7)},
{(4,3), (3,2), (2,7)}, {(4,3), (3,6), (6,7)},

{(5,7)} only for occupants originally located at node 5,
{(6,7)}.

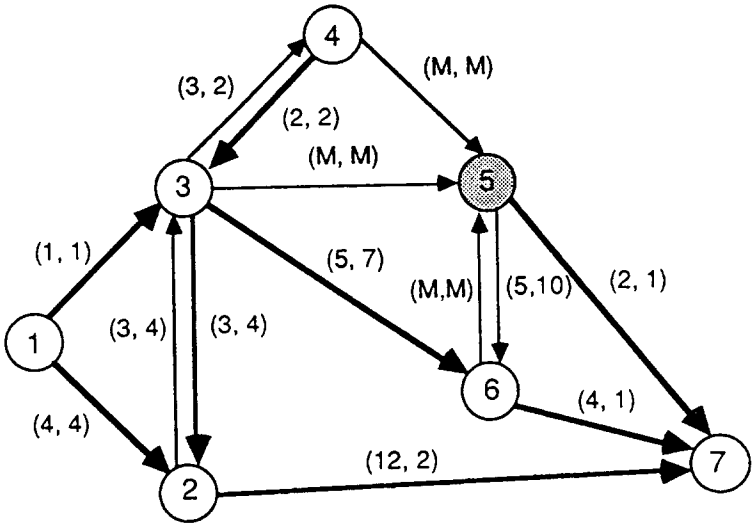


FIGURE 5. Two-attribute network with time dependent costs and non-dominated paths.

Note two new non-dominated links (4,3) and (3,6) that contribute to the new non-dominated paths in the network. The presence of the very costly links to the impassable node 5 has caused a revision to the set of solution paths.

DISCUSSION AND ANALYSIS

Solving the optimal path problems in the preceeding section resulted in different solutions that depend on the concept of optimality assumed for each optimization model.

The models were developed for one network whose nodes and links identify all potential egress routes in the house. Number and location of nodes and links are, of course, arbitrary but should agree with physical locations of occupants and their movement, location of decision points (e.g. doorways) and final exit. Developing a network for the house is the first major phase of the modeling process.

The multi-attribute model allows for simultaneous analysis of several costs that illustrate the multiple objective nature of the evacuation process. Moreover, given non-dominated paths by applying a multi-attribute analysis one can obtain a fresh insight into a building design with respect to the location of a fire.

In order to perform the analysis one can plot non-dominated paths in the two-attribute space. Figure 6 depicts the all non-dominated paths found in the two-attribute network with constant costs and time dependent costs. The origin node number assigned to each non-dominated path helps identify the low and high risk zones within the house. The paths originating at node 2 (of cost (11,13)) and node 1 (of cost (16,6)) belong to the high risk zone due to the large values of time and distance to travel. The paths from node 5 and 6 are, of course, in the lowest risk zone. When node 5 becomes impassable, the path from node 6 stays in the lowest risk zone, but the only two paths leaving node 4 (of cost (11,10) and

(17,8)) are in the high risk zone. Thus, one may conclude that a fire in the dining area of the house makes the evacuation from the kitchen very difficult.

Assuming that one node at a time becomes impassable, we can construct multi-attribute graphs, as in Figure 6, for each node of the network. We can then determine the high and low risk zones in the house for any location of the fire. Such a classification could have a new impact on the design of residential houses in terms of structure, arrangement, and location of rooms and hallways. Furthermore, occupants' decisions and actions to be taken during the evacuation could be motivated by the study of the high and low risk zones in the house.

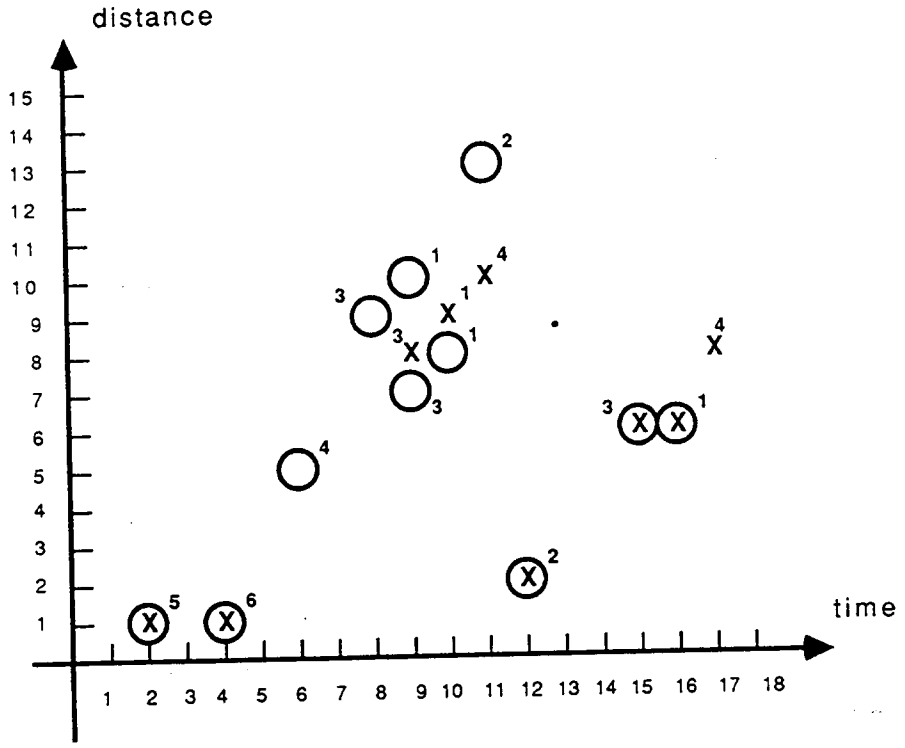


FIGURE 6. Costs of non-dominated paths for two-attribute network with constant costs (○) and time dependent costs (X) with origin node numbers.

CONCLUSIONS

A mathematical optimization approach provided by dynamic programming allows us to simultaneously find optimal egress paths for all occupants in a residential building. Time varying network attributes, as well as multiple attributes on each link of a network, may be handled in the same framework. Decision making in a dynamic environment which includes conflicting goals more faithfully reflects the situation faced by the occupants of a residential building which is involved in a fire.

From the optimal solutions we may gain several insights not available in earlier

analyses. Revised paths, which may be quite instructive, are obtained. The amount of time or distance by which the optimal paths increase when a room becomes impassable may be computed. All non-dominated paths for the building may be plotted in multi-attribute graphs which allow for classification and a new way to view a building.

The optimization models developed in this paper are of special interest for the Center for Fire Research (CFR) at the National Institute of Standards and Technology. The dynamic programming approach relates very well to EXITT [3], [4], and HAZARD I [5], [6], developed at CFR and so there is a possibility of incorporating this methodology in the simulation models of CFR.

ACKNOWLEDGEMENT

This research was supported by Grant No 60NANB0D1023 from the Center for Fire Research, National Institute of Standards and Technology, Gaithersburg, Maryland, U.S.A.

REFERENCES

1. Berlin, G. N., Dutt, A. and Gupta, S. M., "Modeling Emergency Evacuation from Group Homes," Fire Technology, 18, 38-48, 1982.
2. Stahl, F. I., "BFIRES-II: A Behavior Based Computer Simulation of Emergency Egress During Fires," Fire Technology , 18, 49-65, 1982.
3. Levin, B. M., "EXITT - A Simulation Model of Occupant Decisions and Actions in Residential Fires: User's Guide and Program Description," NBSIR 87-3591, Center for Fire Research, National Bureau of Standards, Gaithersburg, Maryland, 1987.
4. Levin, B. M., "EXITT - A Simulation Model of Occupant Decisions and Actions in Residential Fires," in Fire Safety Science -Proceedings of the Second International Symposium, International Association of Fire Safety Science, Tokyo, June 13-17, 1988, Hemisphere Publishing Company, New York, 1989.
5. Bukowski, R. B., Peacock, R. D., Jones, W. W. and Forney, C. L.,"Technical Reference Guide for HAZARD I, Fire Hazard Assessment Method," Handbook 146, Volume II, U. S. National Institute of Standards and Technology, Gaithersburg, Maryland, 1989.
6. Peacock, R. D. and Bukowski, R. W., "A Prototype Methodology for Fire Hazard Analysis," Fire Technology, 26, 15-40, 1990.
7. Jones, W. W. and Peacock, R. D., "Refinement and Experimental Verification of a Model for Fire Growth and Smoke Transport", in Fire Safety Science - Proceedings of the Second International Symposium, International Association of Fire Safety Science, Tokyo, June 13-17, 1988, Hemisphere Publishing Company, New York, 1989.
8. Kisko, T. M. and Francis, R. L., "EVACNET+: A Computer Program to Determine Optimal Building Evacuation Plans," Fire Safety Journal , 9, 220, 1985.
9. Kostreva, M. M. and Wiecek, M. M., "Time Dependency in Multiple Objective Dynamic Programming," Department of Mathematical Sciences, Clemson University Technical Report #601, Clemson, South Carolina, 1991.
10. Bellman, R., "On a Routing Problem," Quarterly of Applied Mathematics, 16, 87-90, 1958.
11. Cooke, K. L. and Halsey, E., "The Shortest Route Through a Network with Time-Dependent Internodal Transit Times," Journal of Mathematical Analysis and Applications, 14, 493-498, 1966.
12. Bellman, R. and Kalaba, R., Dynamic Programming and Modern Control Theory, Academic Press, New York, 1965.
13. Daellenbach, H. G. and De Kluyver, C. A., "Note on Multiple Objective Dynamic Programming," Journal of the Operational Research Society, 31, 591-594, 1980.

Appendix 2

"Multicriteria Decision Making in Fire Egress Analysis," Preprints of IFAC/IFORS Workshop on Support Systems for Decision and Negotiation Processes, (R. Kulikowski, Z. Nahorski, J. Owsinski, and A. Straszak, eds.) Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland, June 1992, pp. 285-290 (with M. Wiecek).

SUPPORT SYSTEMS FOR DECISION AND NEGOTIATION PROCESSES
Preprints. IFAC/IFORS/IIASA/TIMS Workshop, June 24-26, 1992, Warsaw, Poland

MULTICRITERIA DECISION MAKING IN FIRE EGRESS ANALYSIS

Michael M. Kostreva
Malgorzata M. Wiecek

Department of Mathematical Sciences
Clemson University
Clemson, SC 29634
U. S. A.

Abstract: This paper presents a multicriteria dynamic approach to fire egress analysis. An algorithm for generating all nondominated egress paths for human occupants in a fire building is given. The approach uses the most recent theoretical developments in time dependent vector dynamic programming. An interesting application related to a fire in a North Carolina food plant is shown.

Keywords: vector dynamic programming, networks, egress models, multicriteria decision making.

1. Introduction

In this research the movements of human occupants of a building who are reacting to the presence of a fire in the building are considered. Such movement is generally known as egress and thus the present study is concerned with fire egress analysis. Our work seems to be the first to apply multi-objective programming models to this problem. The motivation to introduce more than one objective function to be minimized is quite natural in the fire environment. A building fire contains multiple hazards to be avoided by simultaneously minimizing travel time, distance of travel, amount of smoke inhaled, amounts of toxins encountered, etc. It is the goal of fire egress analysis to find all nondominated paths for each such occupant and then to make further study of the paths and their corresponding cost vectors. By obtaining such a detailed solution one obtains a deeper understanding of the building structure, the occupants and how they might best manage the risk of a fire.

A building fire and the reaction of the occupants to the fire form an inherently dynamic phenomenon. As such, the data to be considered in decision making is comprised of functions of time which may be derived either from actual fire measurements or from the output of mathematical models of fire and smoke dynamics. At the Building and Fire Research Laboratory of the National Institute of Standards and Technology in Gaithersburg, Maryland, research on the physics of fire and the mathematical modeling of fires in buildings is well established. Recently the Fire Hazards

Analysis group has published a computer package called HAZARD I, (Bukowski et al., 1989), which is comprised of two modules. First, a differential equations model of the spread of flames, chemicals and smoke resulting from a user defined fire in a building is solved by numerical integration. Next, a network is superimposed on the floor plan and the occupants behavior is simulated by a set of heuristic decision rules. The occupants move around in the network until they find an egress path by which they leave the building. The output data from the first module is available as input for the simulation of people-fire interaction.

The research of this paper is designed to provide additional insight into the people-fire interaction within a building. Theoretical foundation for this research work is given by vector dynamic programming (VDP). Hartley (1985), Corley and Moon (1985), and others considered vector routing problems in networks with constant vector costs on links and presented dynamic programming (DP) based algorithms for generating the entire set of nondominated (Pareto-optimal) paths in the network. Cooke and Halsey (1966) were the first who introduced a dynamic network assuming that travel times on links were general functions of time. They developed a DP based algorithm for finding the set of paths from every node to the destination node with shortest travel time.

Different applications have given rise to conducting research on path planning problems in dynamic networks. Orda and Rom (1990), motivated by related problems in computer communication networks, focused on the shortest path problem in the dynamic network with restricted or unrestricted departure time, and developed algorithms for finding an optimal path between two single nodes. Kaufman and Smith (1990) were interested in transporatation planning in congested road networks and specified conditions under which one can efficiently find the set of optimal paths from the origin node to every other node in the network. Kostreva, Wiecek, and Getachew (1991) applied several network models to fire egress analysis for residential buildings. Kostreva and Wiecek (1991) seem to be the first to introduce a network with vector time dependent costs and developed DP based algorithms for finding the set of nondominated paths. In this paper, as a continuation of the research effort reported above, this new approach will be applied to performing multicriteria decision making in fire egress analysis.

2. Time Dependent Vector Dynamic Programming

In the dynamic programming literature two problem formulations are considered: find all nondominated paths 1) from each node to the destination node or 2) from the origin node to every other node. The former formulation is based on the backward approach to DP while the latter uses the forward approach. An algorithm based on the forward technique will be presented in this paper and its applicability to fire egress analysis will be demonstrated.

Consider a general network, not assumed to be acyclic, that consists of N nodes {1, 2, ... , N} and a set of links (i, j) connecting the nodes. Associated with each link is a vector cost $[c_{ij}(t)]$, where the cost functions (c_{ij}: R+ R^{m+}) are assumed to be positive vector valued functions of time. Let $[c_{ij}(t)]_1$ be the time to travel from node i to node j, given that travel starts at time t. The cost to traverse a path p between two nodes of the network is defined as $[c(p)] = \sum_{(i,j)\in p} [c_{ij}(t)]$. It is assumed that the cost of traveling along a link is a function only of the arrival time at the starting node at the link. This assumption, referred to as the frozen link model, allows for fixing a link cost not at the time of leaving the origin node, but later, at the time of arriving at the starting node of the link and, thus, for updating the link cost according to very recent data about the fire.

Let time t be a continuous variable, that is $t \geq 0$, and let the functions $[c_{ij}(t)]_1$ take any positive value. Let node 1 be the origin node. Assume that the departure time from the origin node is $t = 0$. Finally an assumption is introduced which allows the formulation of the principle of optimality for dynamic multiple objective networks. The assumption seems to be very realistic for any fire egress scenario: part a) states that evacuees may not pass each other during evacuation, and part b) refers to deterioration of evacuation conditions over time.

Assumption For any link (i, j) in the network and all $t_1, t_2 \geq 0$, if $t_1 \leq t_2$, then

a) $t_1 + [c_{ij}(t_1)]_1 \leq t_2 + [c_{ij}(t_2)]_1$, and

b) $[c_{ij}(t_1)]_r \leq [c_{ij}(t_2)]_r$ for all $r \in \{2, ..., m\}$.

Theorem : Principle of Optimality for Dynamic Multiple Objective Networks

Under Assumption a) and b), a nondominated path p, that leaves the origin node at time $t = 0$ and arrives at node j at time t_j, has the property that for each node i lying on this path, a subpath p_1, that leaves the origin node at time $t = 0$ and arrives at node i at time t_i, $t_i \leq t_j$, is nondominated.

By Bellman's principle of optimality and the Theorem , we establish that for $t^{\ell} > 0$ and $t^{n} > 0$:

$$\{[G_j^{\ell}(t^{\ell})],\ \ell = 1,...,N_j\} = \text{VMIN}\ \{\ [G_i^{n}(t^{n})] + [c_{ij}(t^{n})],\ n = 1,...,N_i\ \},\quad j=2,3,...,N,$$

$$\{[G_j^{\ell}(t^{\ell})],\ \ell = 1\} = \{\ 0\ \},$$

where $[G_i^{n}(t^{n})]$ is the vector cost of nondominated path n leaving the origin node at time $t = 0$ and arriving at node j at time t^n. Operation VMIN computes vector costs of nondominated paths in the set whose each element is a vector sum of the vector cost of the nondominated path n leaving the origin node at time 0 and arriving at node i at time t^n, and the cost vector of link (i, j) with the arrival time t^n at node i. For details about the computation see Kostreva and Wiecek (1991).

3. Application

The following is an illustrative example of the type of multi-objective analysis which the new theoretical dynamic programming development permits.

On Tuesday September 3, 1991 the worst industrial accident in the history of North Carolina occurred when a fire started in the Imperial Food Products plant in Hamlet, North Carolina, a small town 75 miles southeast of Charlotte. Twenty-five workers were killed in the fire and over 45 others were injured. Greenville (South Carolina) News reports stated that the plant had several blocked exits and lacked a formal evacuation plan. One of the exit doors was blocked by a truck which was subsequently moved forward by the driver to allow the escape of several employees. A floor plan of the plant was published in the Greenville News on September 5 and is depicted in Figure 1. The description given in the newspaper allowed us to construct the following example.

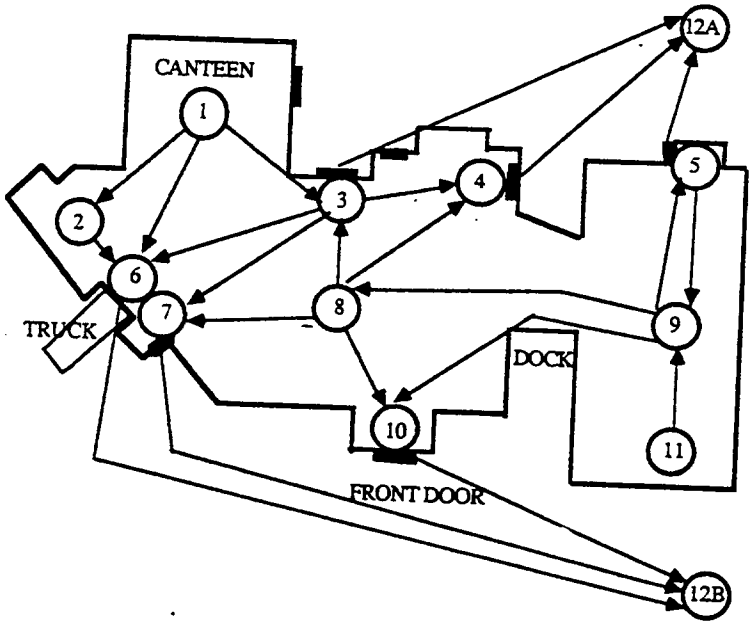


Figure 1. The plant floor plan and related network.

Consider that a fire occurred at the location of node 8 (in the center of the building) and the fire spread rapidly toward node 3. At time t = 3, the associated rooms and passageways become blocked by smoke. Two objectives are considered, travel time and distance of travel. Nodes 12A and 12B represent the same outside location. Assume constant vector costs on the following links: $c_{12} = (2, 2)$, $c_{16} = (3, 2)$, $c_{26} = (1, 1)$ $c_{34} = (1, 1)$, $c_{36} = (3, 2)$, $c_{37} = (2, 2)$, $c_{3,12A} = (2, 2)$, $c_{4,12A} = (2, 2)$, $c_{59} = (1, 1)$, $c_{5,12A} = (1, 1)$, $c_{67} = (1, 1)$, $c_{6,12B} = (2, 2)$, $c_{7,12B} = (2, 2)$, $c_{87} = (1, 1)$, $c_{8,10} = (1, 1)$, $c_{95} = (1, 1)$, $c_{98} = (3, 3)$, $c_{9,10} = (3, 3)$, $c_{10,12B} = (2, 2)$, $c_{11,9} = (1, 1)$, and monotone increasing step functions on the links leading to node 3:

$$c_{13} = \begin{cases} (2, 2), & t < 3 \\ (10, 15), & t \geq 3 \end{cases} \quad \text{and} \quad c_{83} = \begin{cases} (1, 1), & t < 3 \\ (10, 15), & t \geq 3 . \end{cases}$$

Applying the forward approach for every node i in the network, i = 1, ... , 12, we can find the set of all nondominated paths leaving that node at time t = 0 and arriving at node j, $j \neq i$, that is, we find all nondominated paths from node i, i = 1, ..., 11, to the destination node 12. Nondominated evacuation paths from nodes 1 and 9 are of special interest due to these nodes' locations. There are two nondominated paths from node 1 to node 12, namely path {(1,3), (3,12A)} that has the total cost of (4, 5), and path {(1,6), (6,12B)} with the total cost of (5, 4). Path {(9,5), (5,12A)} is the only nondominated path leading from node 9 to node 12 with the total cost of (2, 2).

From these calculations we do not wish to conclude anything in particular about the North Carolina fire. It is simply of interest to know about nondominated evacuation paths in order to prepare the occupants of a building for any emergency which might arise.

4. Conclusions

In this short paper we have introduced a dynamic programming application which has two unusual and important characteristics: multiple criteria on each link of the related network and time dependent cost functions. A principle of optimality is given which allows the computation of the set of all nondominated paths from an origin node to any other node in the network. An illustrative example related to fire egress analysis has been included in which some costs are time dependent discontinuous functions. For this example, all nondominated paths are computed leading from two of the nodes(rooms of a building) to the destination node(outside of the building). The technique makes an assumption which is realistic and quite natural for fire egress analysis and thus it is of general applicability for these problems.

5. Acknowledgement

This research was supported by Grant No 60NANB0D1023 from the Center for Fire Research, U.S. National Institute of Standards and Technology, Gaithersburg, Maryland, U. S. A.

6. References

Bukowski, R. B., Peacock, R. D., Jones W. W., Forney C. L. (1989) *Technical Reference Guide for HAZARD I, Fire hazard Assessment Method*, Handbook 146, Vol. II, U. S. National Institute of Standards and Technology, Gaithersburg, Maryland.

Cooke K.L., Halsey E. (1966) The shortest route through a network with time-dependent internodal transit times, *Journal of Mathematical Analysis and Applications*, 14, 493–498.

Corley H. W., Moon I. D. (1985) Shortest paths in networks with vector weights, *Journal of Optimization Theory and Applications*, 46, 79-86.

Hartley R. (1985) Vector optimal routing by dynamic programming. In: *Mathematics of Multiobjective Optimization*, P. Serafini (ed.), Springer-Verlag, Berlin, 215-224.

Kaufman D. E., Smith R. L. (1990) Minimum Travel Time Paths in Dynamic Networks with Application to Intelligent Vehicle-Highway Systems, IVHS Technical Report #90-11, Transportation Research Institute, University of Michigan, Ann Arbor, MI.

Kostreva M. M., Wiecek M. M., Getachew T., Optimization models in fire egress analysis for residential buildings, to appear in *Proceedings of the Third International Symposium on Fire Safety Science*, Edinburgh, United Kingdom, July 8-12, 1991.

Kostreva M. M., Wiecek M. M., (1991) Time Dependency in Multiple Objective Dynamic Programming, Technical Report #601, Department of Mathematical Sciences, Clemson University, Clemson, SC.

Orda A., Rom R., (1990) Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length, *Journal of the Association for Computing Machinery*, 32, no.3, 607-625.

Appendix 3

"An Algorithm for Multiple-Objective Path Optimization with Time Dependent Links," Proceedings of 10th International Conference on Multiple Criteria Decision Making, Volume 3, pp. 319- 330, Taipei, Taiwan, July 1992 (with T. Getachew).

AN ALGORITHM FOR MULTIPLE-OBJECTIVE PATH OPTIMIZATION WITH TIME DEPENDENT LINKS

Teodros Getachew
Michael Kostreva
Mathematical Sciences Department
Clemson University
Clemson, South Carolina 29634-1907
USA

Abstract.

The applicability of shortest-path algorithms is well known. There have been two separate extensions of the classical shortest-path problem. The first is an extension to finding optimal paths in networks whose links have time-dependent costs, while the second is concerned with the establishment of algorithms to find all non-dominated paths through a network with vector-valued costs. Recent work has shown that the adequate analysis of some important problems, characterized by multiple-objectives in a time-dependent context, requires the formulation of models and algorithms that are capable of capturing bothof these advances.

We present an algorithm that solves the following problem. Let a network whose links have vector-valued, time-dependent costs be given. Suppose a distinguished node, called the destination node is selected. Find all non-dominated, paths from all nodes to the destination node. Apart from satisfying a certain boundedness condition, the cost functions are not restricted. The algorithm is recursive, finite, and constructive. It is a backward and forward procedure, operating simultaneously in link-space and in path-space. No time grid is required. It reduces to Multi-Objective Dynamic Programming in the case of constant costs. In general however, this does not hold, as the Principle of Optimality may be violated. We present computational experience for piecewise linear cost-functions.

Introduction.

The problem of interest is the following: given a set of points, called nodes, joined by edges, called links, with each link having associated with it a transition cost function of time, that is, a function that specifies the cost of making the transition between the node that defines one end of the link to the node that defines the other, as a function of arrival time at the inital node of the link; among those continuous, non self-crossing paths from a given node to a specially designated node, called the destination node, which have the least cost ?

The main source of the difficulty in this problem lies in the fact that whereas paths have unique costs, links do not. It is to be recalled that in classical Dynamic Programming, one only considers links whose costs are time-invariant, and in particular, independent of the paths containing the links. In this, the time-variant case however, the cost of a link is not only a function of the link itself but also of the path which contains that the link. Thus a link may have several costs, each corresponding to a different path which contains it.

Definition: Let $\mathfrak{F}$ be a set of functions with domain $\mathfrak{R}^{+} \cup \{0\}$ and range $\mathfrak{R}^{+}$, bounded above and below on bounded intervals.
Then the link-transition cost functions are given by the range of the function $\mathfrak{T}$ where:

$$\mathfrak{T} : \mathfrak{L} \rightarrow \mathfrak{F}, \text{ given by } \mathfrak{T}((i,j)) = \mathfrak{T}_{ij}(t) \in \mathfrak{F}.$$

The cost on a link, as noted above is a function, not only of time, but also of the paths to which it belongs. Given a particular path, and a particular link on it, the cost on the link is determined by the time of arrival, via the path, at the initial node of the link. This is formalized in terms of the arrival function.

Definition: Let $\mathcal{P} = \{(i, j_1), (j_1, j_2), \ldots, (j_{k-2}, j_{k-1}), (j_{k-1}, d)\}$ be a path in $\mathbb{P}_i^k(\mathcal{G})$. The arrival function $\mathcal{A}: \mathbb{P}(\mathcal{G}) \times \mathfrak{N} \rightarrow \mathfrak{R}^{+}$ is defined recursively on initial link-nodes as follows:

$$\mathcal{A}(\mathcal{P}, i) = 0;$$

- Suppose $\mathcal{A}(\mathcal{P}, j_{k-1})$ has been defined for $r \leq s-1$; let $(j_s, j_{s+1}) \in \mathcal{P}$.

$$\mathcal{A}(\mathcal{P}, j_{s+1}) = \mathcal{A}(\mathcal{P}, j_s) + \mathfrak{T}_{j_s j_{s+1}}(\mathcal{A}(\mathcal{P}, j_s)).$$

Definition: $C: \mathfrak{L} \rightarrow \mathfrak{T} \times \mathfrak{T} \times \ldots \times \mathfrak{T}$ such that,

$$C(i,j) = (c_{ij}^{(1)}(t), \ldots, c_{ij}^{(p)}(t)),$$

where each $c_{ij}^{(k)} \in \mathfrak{F}$.

The "path costing" functions Γ_o and Γ are defined next:

Defintion: Let

$$c_o^{(k)}(n_i, n_j) = \inf_{t \in I(n_i, n_j)} c_{n_i n_j}^{(k)}(t) \text{ where,}$$

$$I(n_i, n_j) = [\alpha, \beta],$$

with

$$\alpha = \min\ \mathcal{C}(\mathcal{P}, n_i) \text{ and } \beta = \max\ \mathcal{C}(\mathcal{P}, n_j)\ ,$$

$$(n_i\ , n_j\ ) \in \mathcal{P}.$$

Now, let $\mathcal{P} \in \mathbb{P}_i\ (\mathcal{G})$ be of cardinality k.

Then Γ_o: $\mathbb{P}(\mathcal{G}) \rightarrow \mathcal{R}^+ \times \mathcal{R}^+ \times \ldots \times \mathcal{R}^+$ is defined by

$$\Gamma_o(\mathcal{P}) = \vec{c}_o\ \ (i, j_1) + \vec{c}_o\ \ (j_{k-1}, d) + \sum_{s=1}^{k-2} (\vec{c}_o\ \ (j_s, j_{s+1})),$$

where,

$$\vec{c}_o\ \ (j_q, j_s) = (c_o^{(1)}(j_q, j_s), \ldots,\ c_o^{(k)}(j_q, j_s)).$$

This is the definition of the forward costing function:

Γ: $\mathbb{P}(\mathcal{G}) \rightarrow \mathcal{R}^+ \times \mathcal{R}^+ \times \ldots \times \mathcal{R}^+$ is defined by,

$$\Gamma(\mathcal{P}) = \vec{c}_{ij_1}\ (0) + \vec{c}_{j_{k-1}d}(\mathcal{C}(\mathcal{P}, j_{k-1}))$$

$$+ \sum_{s=1}^{k-2} (\vec{c}_{j_s j_{s+1}}(\mathcal{C}(\mathcal{P}, j_s)))\ .$$

Finally, the notion of efficiency is formally stated in terms of the notation given above.

Definition: Let $\mathcal{P} \in \mathbb{P}^{(k)}(\mathcal{G})$. Then,

$\mathcal{P} \in \mathcal{E}ff_i^{(k)}\ (P_o)$ if and only if the set

$\{\mathcal{P}'\colon \mathcal{P}' \in \mathbb{P}^{(k)}(\mathcal{G})$ and $\Gamma_o(\mathcal{P}') \leq \Gamma_o(\mathcal{P})$ and $\Gamma_o(\mathcal{P}') \neq \Gamma_o(\mathcal{P})\}$ is empty.

Similarly, $\mathcal{P} \in \mathcal{E}ff_i^{(k)}(P)$ if and only if the set

$\{\mathcal{P}'\colon \mathcal{P}' \in \mathbb{P}^{(k)}(\mathcal{G})$ and $\Gamma(\mathcal{P}') \leq \Gamma(\mathcal{P})$ and $\Gamma(\mathcal{P}') \neq \Gamma(\mathcal{P})\}$ is empty.

Note that $\mathcal{E}ff(P_o) = \mathcal{E}ff_i^{(n-1)}(P_o)$ and $\mathcal{E}ff(P) = \mathcal{E}ff_i^{(n-1)}(P)$.

The algorithm:

The algorithm that is presented below has the following features:

i) It is a recursive algorithm in which only the information generated in a previous iteration is used in the execution of a subsequent one;
ii) It takes place both in the space of links, in a backward Dynamic Programming phase, and the space of paths, in a forward integration phase.

Iteration I_o:

Given a network, with each link having an associated transition cost function as defined above, the algorithm begins with the initial partition of the set of paths; namely, the partition consisting of the empty set and the entire set of paths with all links costed as in Γ_o.

Note that this costing yields a unique network, as each link transition cost function has a unique infimum. What follows then is an application of backward Dynamic Programming to this network.

$$F_{0i}^{(1)} = \text{VMIN}\{\Gamma_o(\mathcal{P}) : \mathcal{P} \in \mathbb{P}_i^1(\mathcal{G})\};$$

$$F_{0i}^{(2)} = \text{VMIN}\{\Gamma_o(\mathcal{P}): \Gamma_o(\mathcal{P}) = \vec{c}_o(i,j) + f_{0j}^{(1)},\ f_{0j}^{(1)} \in F_{0j}^{(1)}\};$$

$$F_{0i}^{(r)} = \text{VMIN}\{\Gamma_o(\mathcal{P}): \Gamma_o(\mathcal{P}) = \vec{c}_o(i,j) + f_{0j}^{(r-1)},\ f_{0j}^{(r-1)} \in F_{0j}^{(r-1)}\};$$

$$F_{0i}^{(n-1)} = \text{VMIN}\{\Gamma_o(\mathcal{P}): \Gamma_o(\mathcal{P}) = \vec{c}_o(i,j) + f_{0j}^{(n-2)},\ f_{0j}^{(n-2)} \in F_{0j}^{(n-2)}\};$$

The backward Dynamic Programming stage terminates with the set $F_{0i}^{(n-1)}$.

Let,

$$F_{0i} \equiv F_{0i}^{(n-1)},$$

and

$$\mathbb{P}(F_{0i}) = \{\mathcal{P} : \mathcal{P} \in \mathbb{P}(\mathcal{G}) \text{ and } \Gamma_o(\mathcal{P}) \in F_{0i}\}.$$

The true cost of each element whose cost is in the set F_{0i} is now found by the evaluation, via the costing function Γ. If this cost differs from its cost under Γ_o, it is added to the set V_{0i}. Let

$$V_{0i} = \{\mathcal{P}: \ \mathcal{P} \in \mathbb{P}(F_{0i}) \text{ and } \Gamma_o(\mathcal{P}) \leq \Gamma(\mathcal{P}) \text{ and } \Gamma_o(\mathcal{P}) \neq \Gamma(\mathcal{P})\}.$$

If V_{0i} is empty, then

$$\text{STOP; } \mathbb{P}(F_{0i}) = \mathfrak{C}\, ff_i(P);$$

else let,

$$\mathcal{U}_{0i} = \varnothing$$

and

$$\mathcal{U}_{1i} = V_{0i} \cup \mathcal{U}_{0i}.$$

Suppose now that I_{r-1} and $\mathcal{U}_r$ have been defined. Then, define

Iteration I_r:

The set S_r:consisting of all subpaths of all the elements of the set $\mathcal{U}_r$, central to the definition of iteration I_r, is defined next.

Definition: The set S_r of forbidden paths is defined by the equation:

$$S_r = \{\mathcal{P}: \mathcal{P} \in \mathbb{P}(\mathcal{G}) \text{ such that } \mathcal{P} \text{ is a subpath of } \mathcal{P}', \mathcal{P}' \in \mathcal{U}_r\}.$$

This set defines the partition and the costing for iteration I_r.The path-costing function for the rth iteration is now defined by:

Definition: $\Gamma_r: \mathbb{P}(\mathcal{G}) \rightarrow \mathcal{R}^+ \times \mathcal{R}^+ \times \ldots \times \mathcal{R}^+$,

where,

$$\Gamma_r(\mathcal{P}) = \Gamma_0(\mathcal{P}) \text{ if } \mathbb{P} \notin S_r,$$

$$\Gamma(\mathcal{P}) \text{ otherwise.}$$

Let

$$S_{ri} \equiv \mathbb{P}_i(\mathcal{G}) \cap S_r,$$

and

$$S_{ri}^k \equiv \mathbb{P}_i^k(\mathcal{G}) \cap S_r.$$

The backward Dynamic Programming is applied to the set of paths that are in the complement of the set S_{ri}.

$$F_{ri}^{(1)} = \text{VMIN}\{\Gamma_r(\mathcal{P}) : \mathbb{P} \in \mathbb{P}_i^1(\mathcal{G}), \text{ and } P \notin S_{ri}\};$$

Paths of length at least two in the complement of S_{ri} can take two forms: those all of whose links but the latest come from the previous vector-minimization, and those all whose links except the latest come from a set S_{rj}. Accordingly, vector-minimization over sets of paths of cardinality greater than one must involve, not only paths from the preceding vector-minimization step, but also paths from appropriate forbidden sets.

$$F_{ri}^{(2)} = \text{VMIN}\{\{\Gamma_r(\mathcal{P}): \Gamma_r(\mathcal{P}) = f_{rj}^{(1)} + \vec{c}_0(i,j) \ , \ f_{rj}^{(1)} \in F_{rj}^{(1)} \ , p \notin S_{ri}\} \cup$$

$$\{\Gamma_r(\mathcal{P}) : \Gamma_r(\mathcal{P}) = \Gamma_0(\mathcal{P}') + \vec{c}_0(i,j) \ , \ p' \in S_{rj}^1 \text{ and } p \notin \Gamma_{ri}^0\}\}.$$

The backward Dynamic Programming ends with the evaluation of the set $F_{ri}^{(n-1)}$.

$$F_{ri}^{(n-1)} = \text{VMIN}\{\{\Gamma_r(\mathcal{P}): \Gamma_r(\mathcal{P}) = f_{rj}^{(n-2)} + \vec{c}_0(i,j) \ , \ f_{rj}^{(n-2)} \in F_{rj}^{(n-2)} \ ,$$

$$p \notin S_{ri}\} \cup \{\Gamma_r(\mathcal{P}) : \Gamma_r(\mathcal{P}) = \Gamma_0(\mathcal{P}') + \vec{c}_0(i,j) \ , \ p' \in S_{rj}^{n-2} \text{ and } p \notin \Gamma_{ri}^0\}\}.$$

Let $F_{ri} = \text{VMIN}\{F_{ri}^{(n-1)} \cup \{S_{ri}\}\}$.

where

$$\{S_{ri}\} \equiv \{\Gamma_r(\mathcal{P}): p \in S_{ri}\}.$$

Letting,

$$P(F_{ri}) \equiv \{ p: p \in P_g^i \text{ and } \Gamma_r(\mathcal{P}) \in F_{ri}\},$$

the set V_{ri} is defined to be the set of paths whose cost under the costing Γ_r differs from that under Γ.

Definition: $V_{ri} = \{\mathcal{P}: \mathcal{P} \in \mathbb{P}(F_{ri}) \text{ and } \Gamma_r(\mathcal{P}) \leq \Gamma(\mathcal{P}) \text{ and } \Gamma_r(\mathcal{P}) \neq \Gamma(\mathcal{P})\}$.

If V_{ri} is empty,

$$\text{STOP; } \mathbb{P}(F_{ri}) = \mathcal{E}ff_i(P)$$

else let,

$$\mathcal{U}_{(r+1)i} \equiv V_{ri} \cup \mathcal{U}_{ri} \; ; \; \mathcal{U}_{r+1} \equiv \bigcup_{i \neq d} \mathcal{U}_{(r+1)i}$$

Theoretical Results.
Note: In the sequel, the symbol $\leq\!\!\neq$ shall stand stand for "at most as large as, vector-wise, but distinct from".
The following theorems have been established (See [5] for more details).

Theorem 1: The algorithm terminates after a finite number of iterations. Let

$$\mathbb{P}(\mathcal{G}_r') \equiv \mathbb{P}(\mathcal{G}) \backslash S_r,$$

and

$$\mathbb{P}_i(\mathcal{G}_r') \equiv \mathbb{P}(\mathcal{G}_r') \cap \mathbb{P}_i(\mathcal{G})$$

Define $\mathcal{E}ff(\mathbb{P}_i(\mathcal{G}_r'))$ by the condition $\mathcal{P} \in \mathcal{E}ff(\mathbb{P}_i(\mathcal{G}_r'))$ if and only if the set

$$\{ \mathcal{P}' : \mathcal{P}' \in \mathbb{P}_i(\mathcal{G}_r') \text{ and } \Gamma_0(\mathcal{P}') \leq \neq \Gamma_0(\mathcal{P}) \} \text{ is empty.}$$

We then have,

Theorem 2: $\mathbb{P}(F_{ri}^{n-1}) = \mathcal{E}ff(\mathbb{P}_i(\mathcal{G}_r'))$.

Define the set $\mathcal{E}ff_i(P_r)$ by:

Definition: $\mathcal{P} \in \mathcal{E}ff_i(P_r)$ if and only if the set $\{\mathcal{P}' : \mathcal{P}' \in \mathbb{P}_i(\mathcal{G}) \text{ and } \Gamma_r(\mathcal{P}')$

$\leq \neq \Gamma_r(\mathcal{P})\}$ is empty.

Theorem 3: $\mathbb{P}(F_{ri}) = \mathcal{E}ff_i(P_r)$.

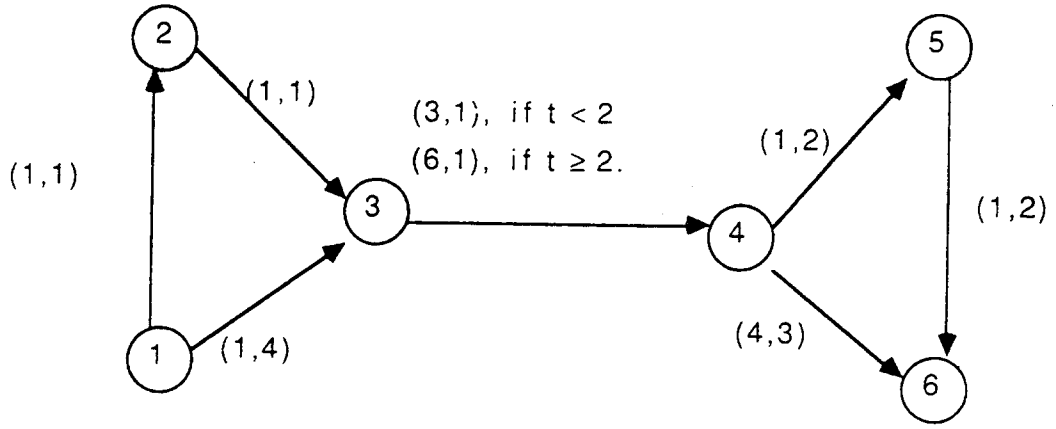
In the next theorem, the main theoretical result of this dissertation is established; namely that after a finite number of iterations, the algorithm determines the entire set of efficient paths, for all possible initial nodes.

Theorem 4: Let the algorithm terminate after t iterations. Then,

$$\mathbb{P}(F_{ti}) = \mathcal{E}ff_i(P).$$

A Numerical Example.

The following directed network with node six as the destination node shall be used to illustrate the algorithm. The first component of the cost vectors represents transit times, the second represents distance.



$F_{01}^{(5)}$ = VMIN{(8,8) , (9,6) , (6,9) , (1,1) + (6,6)}= {(8,8) , (9,6) , (6,9) , (7,7)}

$F_{02}^{(5)}$ = VMIN{(8,5) , (1,1) + (5,5)} = {(8,5) , (6,6)};

$F_{03}^{(5)}$ = VMIN{(7,4) , (3,1) + (2,4)}= {(7,4) , (5,5)};

$F_{04}^{(5)}$ = VMIN{(4,3) , (1,2) + (1,2)}= VMIN{(4,3) , (2,4)}= {(4,3) , (2,4)};

$F_{05}^{(5)}$ = VMIN{(1,2)} = {(1,2)}.

Forward costing shows that,

$$V_{02} = V_{03} = V_{04} = V_{05} = \varnothing$$

Therefore,

$$\mathfrak{C}\, ff_i(P) = F_{0i}^{(5)},$$

for i = 2 , 3 , 4 , 5.
Thus the algorithm continues for node one only.

We have,

V_{01} = {(12346) , (123456)};

$U_{11} = V_{01}$;

S_1 = {(46),(346),(2346),(12346),(56),(456),(3456),
(23456),(123456)};

and

$\Gamma_1(\mathcal{P}) = \{\Gamma_0(\mathcal{P})$ if $\mathcal{P} \notin S_1$, and $\Gamma(\mathcal{P})$ otherwise$\}$.

$F_{11}^{(1)} = \varnothing$; $F_{11}^{(2)} = \varnothing$;

$F_{11}^{(3)}$ = VMIN {(1,4) + (7,4) } = {(8.8)};

$F_{11}^{(4)}$ = VMIN {(8,8) , (1,4) + (5,5)} = {(8.8) , (5,9)};

$F_{11}^{(5)} = F_{11}^{(6)} = F_{11}^{(4)}$;

F_{11} = VMIN { (8,8) , (9,5) , (10,7) , (12,6)}
= { (8,8) , (9,5) , (10,7) , (12,6) }.

$V_{11} = \varnothing$. So, the algorithm terminates.

The efficient paths from the given node to the destination node are:
Node 1: { (123456), (12346), (13456), (1346) };
2: { (23456), (2346) };
3: { (3456) , (346) }
4: { (456) , (46) };
5: { (56) }.

Bibliography.

1. R. Bellman, On a routing problem, Quarterly of Applied Mathematics 16 (1958), 87-90.
2. J. Brumbaugh-Smith and D.Shier, An empirical investigation of some bicriterion shortest path algorithms, European Journal of Operational Research 43 (1989) 216-224.
3. K. L. Cooke and E. Halsey, The shortest route through a network with time-dependent internodal transit times, Journal of Mathematical Analysis and Applications 14 (1966), 493-498.
4. H. W. Corley and I. D. Moon, Shortest paths in networks with vector weights, Journal of Optimization Theory and Applications 46 (1985), 79-86.
5. T. Getachew, An algorithm for multiple-objective network optimization with time variant link-costs, Ph.D. Thesis, Clemson University, 1992.
6. D. E. Kaufman and R. L. Smith, Minimum travel time paths in dynamic networks with application to intelligent vehicle- highway systems, University of Michigan, Transportation Research Institute, IVHS Technical Report-90-11, 1990.
7. M. M. Kostreva, M. M. Wiecek and T. Getachew, Optimization models in fire egress analysis for residential buildings, appear in Proceedings of the Third International Symposium on Fire Saftey Science, Edinburgh, United Kingdom, July 8-12, 1991, 805-814.
8. M. M. Kostreva and M. M. Wiecek, Time dependency in multiple-objective dynamic programming, Department of Mathematical Sciences, Clemson University, Technical Report #601, 1991, to appear in Journal of Mathematical Analysis and Applications.
9. J. Mote, Ishwar Murthy and David L. Olson, A parametric approach to solving bicriterion shortest path problems, European Journal of Operational Research, 53 (1991) 81-92.
10. A. Orda and R. Rom, Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length, Journal of the Association for Computing Machinery 37 (3) (1990), 607-625.
11. A. B. Philpott, A class of continous-time shortest-path problems, to appear in SIAM Journal on Control and Optimization.

Appendix 4

"Approximation in Time Dependent Multiple Objective Path Planning," Proceedings of 1992 IEEE International Conference on Systems, Man and Cybernetics, Volume 1, pp. 861-866, Chicago, Illinois, October 1992 (with M. Wiecek).

Approximation in Time Dependent Multiple Objective Path Planning

Michael M. Kostreva
Malgorzata M. Wiecek
Department of Mathematical Sciences
Clemson University
Clemson, SC 29634-1907

Abstract - **This paper presents a new method for approximating the set of nondominated paths in a dynamic network with vector costs that are monotone increasing step functions of time. The method includes solving a sequence of constant cost dynamic programming problems. Some illustrative examples are provided.**

I. INTRODUCTION

The problem of planning paths in a multiple objective dynamic network arises in transportation, telecommunications, fire egress analysis, and many other areas. Such a network comprehends vector time-dependent cost functions on links. Although such problems have been known for some time, recent progress has been substantial [2-5, 9] with fire egress analysis providing the motivating application area, and dynamic programming the theoretical framework. Drawing on classical works such as [1] as well as newer papers such as [5,6,8] and monographs such as [7], some extensions of classical dynamic programming have been developed [3] and applied [2-4], and some algorithms [9] which are powerful as well as novel have been derived. One issue which has not been considered is the use of simplified approximation methods to obtain partial descriptions of the Pareto optimal paths. In the earlier approaches, all such paths have been computed, stored and analyzed. Here we relax this requirement to study methods which may not compute all such paths and/or may compute paths which are not Pareto optimal. The requirements of solving large networks or real-time applications may necessitate such methods.

In this paper, the cost functions are monotone increasing step functions on $[0, \infty)$, which seem to be very important in applications. These step functions may be used to model the 'failure' of part of the network. For example, in fire egress analysis when a room is suddenly too full of smoke to be traversed, one may represent the fullness by assignment of a large positive number to the travel times, distances, and other cost functions on links entering the node which represents the smoky area. Similarly, if a section of highway becomes blocked because of an accident, a bridge failure or for some other reason, one may model the blockage by step functions with a large positive cost. There are many other phenomena in networks which may be represented by failure of sections of the network and our analysis is designed to handle a fairly general case.

The structure of the cost functions mentioned above leads to the consideration of a new approximation method for multiple objective path planning. Such structure leads to potential gains in the speed of computation which is quite relevant to future applications in which the systems are large scale. That is, given enough computational power and efficient algorithms, individuals in hotel fires may be modeled one by one, and an individual automobile may be modeled in a traffic situation, rather than considering more aggregate models in which flows are treated. The situation we envision is one in which a relatively small number of links in the network have step function costs, while the others have constant costs.

In the next section we formulate the time dependent multiple objective path planning problem and introduce the approximating sequence of constant cost problems. Section III includes the proposed method and explains reasons for approximation. Two path planning examples are provided in section IV. Comparison of the method of approximation with other algorithms is given in the concluding section.

II. PROBLEM FORMULATION

The mathematical framework we consider is a general network, not assumed to be acyclic. It consists of a set of nodes $\{1,2,\ldots,N\}$ and a set of links which indicate connections between nodes, i. e. $\{(i_1, i_2),(i_3, i_4),\ldots\}$. The links' directions are indicated by the order of the indices. So, (3,4) is the link from node 3 to node 4, while (4,3) is the link from node 4 to

This research supported in part by Grant No.. 60NANB0D1023 Building and Fire Research Laboratory, National Institute of Standards and Technology, Gaithersburg, MD.

0-7803-0720-8/92 $3.00

node 3. A path from node i_0 to node i_p is a sequence of links $P=\{(i_0, i_1),(i_1, i_2), \ldots ,(i_{p-1}, i_p)\}$ in which the initial node of each arc is the same as the terminal node of the preceding arc in the sequence and $i_0 , \ldots , i_p$ are all distinct nodes. Let Π be the set of all feasible paths in the network which have the form $\{(i_1, i_2),(i_2, i_3),(i_3, i_4),\ldots,(i_{s-1}, i_s)\}$, where $1 \le i_1 , i_s \le N$.

Each link carries one or more attributes (i.e. time to travel, distance to travel, etc.) which we think of as cost functions. The cost (vector) of a link (i,j) applies to all paths which include link (i,j) The cost functions (c_{ij}: $R^+ \rightarrow R^{m+}$) are assumed to be positive vector valued functions of time, and are not assumed to be continuous. Let $[c_{ij}(t)]$ be the vector cost to travel from node i to node j, given that travel starts at time t, where $[c_{ij}(t)]_1$ is the time to travel between these nodes. The cost to traverse a path p in Π is defined to be

$$[c(p)] = \sum_{(i,j) \in p} [\, c_{ij}(t) \,] .$$

A path in Π is a nondominated path if there is no other path p' in Π with $[\, c(p') \,] \le [\, c(p) \,]$ and $[\, c(p') \,]_r < [\, c(p) \,]_r$ for some $r \in \{1,\ldots,m\}$, where symbol $\le$ in the vector inequality denotes $[\, c(p') \,]_r \le [\, c(p) \,]_r$ for $r = 1, \ldots ,m$.

Let $\kappa \ge 1$, be a positive integer. Suppose that $\{t_1,t_2 ,t_3, t_4 \ldots,t_\kappa\}$ is a given set of times of failure. Then the set of links L_i, will have a step function vector cost with a step at t_i for $i = 1, 2, \ldots , \kappa$. These failure times, link sets and their corresponding step cost functions are extremely relevant to the development of the dynamic programming algorithm. On the remaining links it is assumed (in this structured case) that the costs are constants.

The approach is similar to that given in Ibaraki [7] which he calls successive sublimation dynamic programming. It is more general than Ibaraki's approach because it handles the time varying cost functions described above and also it handles two or more objective functions. Specifically, the problem of finding all nondominated paths from all nodes to a specified destination node is now considered. Let (P) denote this problem for the following discussion.

Now we wish to show how to use the structure of problem (P) to build up a finite set of approximating constant cost networks, for which network routing problems (P^j), $j = 0, 1, \ldots, M$, may be solved. Let node N be the destination node. We introduce the following sets defined for $i = 1, 2, \ldots , N-1$:

$\{B_i\}$ - the set of all paths in the network from node i to node N,

$\{\mathrm{Eff}(B_i)\}$ - the set of all nondominated paths from node i to node N for (P),

$\{\mathrm{Eff}^0(B_i)\}$ - the approximation set to $\{\mathrm{Eff}(B_i)\}$.

$\{\mathrm{Eff}(B_i)^{(j)}\}$ - the set of all nondominated paths from node i to node N for (P^j).

For these constant cost problems a recursive definition is convenient:

(P^0) : The constant cost network routing problem with the constant costs all set to the lowest value of their step functions.

(P^{j+1}) : Modify problem (P^j) as follows:

i) For each nondominated path in $\{\mathrm{Eff}(B_i)^{(j)}\}$, compare its arrival time at the initial node of each of its failure links with that link's failure time. Let $\{M^{(j)}\}$ be the set of links for which arrival at the initial node and passage on the link occur after the failure time.

ii) Introduce the next higher step function costs on links in the set $\{M^{(j)}\}$.

iii) Let $\{\mathrm{Bf}_i^{(j)}\}$ be the subset of nondominated paths $\{\mathrm{Eff}(B_i)^{(j)}\}$ which contain links in $\{M^{(j)}\}$.

Define the accumulation sets of nondominated paths for (P) as follows:

$$\{\mathrm{Eff}^0(B_i)^{(j)}\} = \mathrm{VMIN}\ \{\, [\, \{\mathrm{Eff}(B_i)^{(j)}\} - \{\mathrm{Bf}_i^{(j)}\} \,] \cup \{\mathrm{Eff}^0(B_i)^{(j-1)}\} \,\},$$

where $\{\mathrm{Eff}^0(B_i)^{(-1)}\} = \varnothing$.

Denote this modified problem by (P^{j+1}).

(P^M) : The constant cost network routing problem with all constant costs set equal to the highest values of their step functions.

III. METHOD OF APPROXIMATION

Next the formal statement of the method of approximation is presented. All nondominated paths for problem (P) are of interest. The constant cost routing problems may be solved by either a forward DP approach [5] or a backward DP methodology [6].

METHOD OF APPROXIMATION

STEP 1.

Set j=0. Let $\{\mathrm{Eff}^0(B_i)^{(-1)}\} = \varnothing$ and $\{M^{(-1)}\} = \varnothing$.

STEP 2.

Evaluate the costs for (P^j) introducing the step function costs on links in $\{M^{(j-1)}\}$ and set up the constant cost

network routing problem. Use multiple objective dynamic programming [5, 6] to solve (P^j) by finding $\{Eff(B_i)^{(j)}\}$.

STEP 3.

If j = M, go to STEP 5.

Otherwise, for each nondominated path in $\{Eff(B_i^{(j)})\}$, compare its arrival time at the initial node of each of its failure links with that link's failure time. If all arrive earlier than the failure times, then $\{M^{(j)}\} = \varnothing$ and $\{Bf_i^{(j)}\} = \varnothing$. Then find the set $\{Eff^0(B_i)\} = VMIN \{ \{Eff(B_i)^{(j)}\} \cup \{Eff^0(B_i)^{(j-1)}\} \}$. STOP.

Otherwise, continue to STEP 4.

STEP 4.

Let $\{M^{(j)}\}$ contain those links for which arrival at the initial node and passage on the link both occur after the failure time of the link, and let $\{Bf_i^{(j)}\}$ be the subset of nondominated paths $\{Eff(B_i)^{(j)}\}$ which contain links in $\{M^{(j)}\}$. Find the set $\{Eff^0(B_i)^{(j)}\} = VMIN \{ [\{Eff(B_i)^{(j)}\} - \{Bf_i^{(j)}\}] \cup \{Eff^0(B_i)^{(j-1)}\} \}$.

Set j=j+1 and go to STEP 2.

STEP 5.

Find the set $\{Eff^0(B_i)\} = VMIN \{ \{Eff(B_i)^{(M)}\} \cup \{Eff^0(B_i)^{(M-1)}\} \}$. STOP.

The problem considered here has been solved exactly in terms of dynamic programming in [3] and by a recursive algorithm in [9]. Each of these approaches requires more computation than the method of approximation above. Only fixed dimension, constant cost dynamic programming subproblems are used here unlike earlier approaches which use time grids, individual node restarts, and forward-backward recursive features. The present approach was developed with the motivation that every nondominated path for problem (P) must be associated with some constant cost network problem. However, we do not enumerate all such problems. Hence, the approximation arises.

Although there are some similarities, the method does not utilize the successive sublimation concept [7] because states are not explicitly eliminated. The approximating subproblems are solved by repeated application of the VMIN operation which removes dominated states during the solution of each (P^j). Ibaraki's (single objective) successive sublimation dynamic programming works by changing the state space, whereas our state space remains the same over all (P^j) subproblems. In contrast, our objective function changes as we move from one subproblem (P^j) to its successor, while Ibaraki's method never alters the objective function.

IV. EXAMPLES

In this section we apply the method of approximation to two bi-criteria dynamic networks.

The first example, shows that the final solution set obtained while applying the approximation method does not always include all nondominated paths of the original problem. Consider the network depicted in Figure 1 with one step cost function on link (3,4) of the form:

$c_{34}(t) = \{[3,1], t < 2, [6,2], t \geq 2\}$.

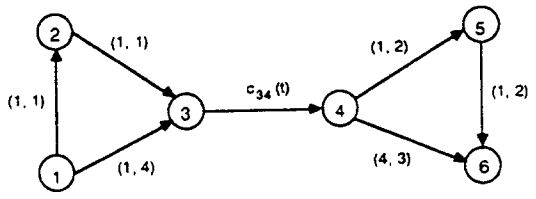


Fig. 1 Bi-criteria network with one step function.

Thus $\kappa = 1$ and $L_1 = (3,4)$. This network was originally developed in [9]. The set of all nondominated paths from node 1 to node 6, includes the following paths

$\{Eff(B_1)\} = \{ \{(1,2), (2,3), (3,4), (4,6)\}, \{(1,3), (3,4), (4,5), (5,6)\}, \{(1,3), (3,4), (4,6)\} \}$,

that are costed [12,7], [6,9], and [8,8], respectively. Now apply the method of approximation for finding nondominated paths from node 1 to node 6 only. Solving problem (P^0) with $c_{34}(t) = [3,1]$ yields the set of nondominated paths

$\{Eff(B_1)^{(0)}\} = \{ \{(1,2), (2,3), (3,4), (4,5), (5,6)\}, \{(1,2), (2,3), (3,4), (4,6)\}, \{(1,3), (3,4), (4,5), (5,6)\} \}$

costed [7,7], [9,6], and [6,9], respectively. $\{M^{(0)}\} = \{(3,4)\}$ and $\{Bf_1^{(0)}\} = \{ \{(1,2), (2,3), (3,4), (4,5), (5,6)\}, \{(1,2), (2,3), (3,4), (4,6)\} \}$. Thus, the first iteration terminates with the accumulation set of nondominated paths for (P)

$\{Eff^0(B_1)^{(0)}\} = \{ \{(1,3), (3,4), (4,5), (5,6)\} \}$.

Problem (P^1) with $c_{34}(t) = [6,2]$ yields the set of nondominated paths

$\{Eff(B_1)^{(1)}\} = \{(1,2), (2,3), (3,4), (4,5), (5,6)\}, \{(1,2), (2,3), (3,4), (4,6)\}, \{(1,3), (3,4), (4,5), (5,6)\}\}$

costed [7,7], [9,6], and [6,9], respectively. Since j = M, we get the final accumulation set of nondominated paths

$\{Eff^0(B_1)\} = VMIN\{\{Eff(B_1)^{(1)}\} \cup \{Eff^0(B_1)^{(0)}\}\} =$
$\{\{(1,2), (2,3), (3,4), (4,5), (5,6)\}, \{(1,2), (2,3), (3,4), (4,6)\}, \{(1,3), (3,4), (4,5), (5,6)\}\},$

that are costed [10,8], [12,7], and [6,9)], respectively. Note that instead of the path {(1,2), (2,3), (3,4), (4,5), (5,6)} of cost [10,8], the path {(1,3), (3,4), (4,6)} of cost [8,8] should be in the accumulation set of problem (P). However, while solving problem (P^0) in the first iteration, the path of cost [8,8], although it was costed as in problem (P), was dominated by the path of cost [7,7] and hence eliminated from the accumulation set. The path of cost [7,7] stayed in the accumulation set of problem (P), costed as in problem (P^0) but not as in problem (P).

The other example presents a bi-criteria dynamic path planning problem for which the approximation method finds all nondominated paths from every node to the destination node. This example was developed in [3] and its complete set of nondominated paths was found applying one of two algorithms presented there. Those algorithms utilize time dependent multiple criteria dynamic programming and generate complete set of nondominated paths for dynamic networks.

Consider the network depicted in Figure 2

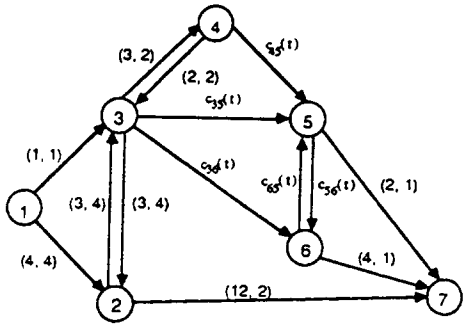


Fig. 2 Bi-criteria network with step cost functions.

and the step cost functions given as:

$c_{35}(t) = \{[6,8], t < 3, [10,12], t \geq 3\},$
$c_{45}(t) = \{[4,3], t < 3, [10,10], t \geq 3\},$
$c_{65}(t) = \{[8,10], t < 3, [10,12], t \geq 3\},$
$c_{36}(t) = \{[5,7], t < 5, [10,12], t \geq 5\},$
$c_{56}(t) = \{[5,10], t < 5, [12,12], t \geq 5\}.$

All other links of the network have constants vector costs as Figure 2 shows. For this network formulate problem (P) as follows: let κ=2 and the failure times be $\{t_1, t_2\} = \{3, 5\}$ with the set of links $L_1 = \{(3,5), (4,5), (6,5)\}$, and $L_2 = \{(3,6), (5,6)\}$.

Problem (P^0) is a constant network routing problem and its solution is depicted in Figure 3.

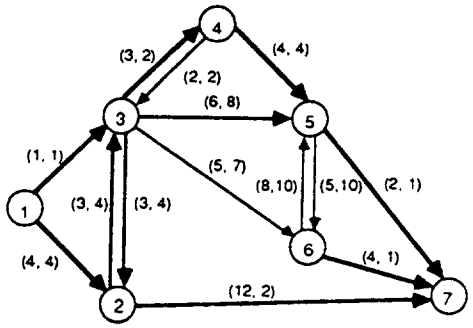


Fig. 3 Problem (P^0) and its nondominated paths.

The sets $\{Eff(B_i)^{(0)}\}$ of all nondominated paths from node i, i = 1, ... , 6, include:

{(1,2), (2,7)}, {(1,3), (3,5), (5,7)}, {(1,3), (3,4), (4,5), (5,7)},
{(2,7)}, {(2,3), (3,5), (5,7)},
{(3,2), (2,7)}, {(3,4), (4,5), (5,7)}, {(3,5), (5,7)},
{(4,5), (5,7)},
{(5,7)},
{(6,7)}.

Notice the arrival times of these nondominated paths at the initial nodes of links (3, 5) and (4, 5) (there is no path going along link (5, 6), (3, 6), and (6,5)). There are three nondominated paths with "late" arrival times, namely: path {(2,3), (3,5), (5,7)} arriving at node 3 at time 3 and traveling along link (3, 5), path {(1,3), (3,4), (4,5), (5,7)} arriving at node 4 at time 4 and traveling along link (4, 5), and path {(3,4), (4,5), (5,7)} arriving at node 4 at time 3 and traveling along link (4, 5). Thus, $\{M^{(0)}\} = \{(3, 5), (4, 5)\}$, and $\{Bf_1^{(0)}\} = \{\{(1,3), (3,4), (4,5), (5,7)\}\}$, $\{Bf_2^{(0)}\} = \{\{(2,3), (3,5), (5,7)\}\}$, $\{Bf_3^{(0)}\} = \{\{(3,4), (4,5), (5,7)\}\}$, and $\{Bf_i^{(0)}\} = \varnothing$ for i = 4, 5, 6. Then the accumulation sets of nondominated paths for (P) are given as:

{(1,2), (2,7)}, {(1,3), (3,5), (5,7)},
{(2,7)},
{(3,2), (2,7)}, {(3,5), (5,7)},
{(4,5), (5,7)},
{(5,7)},
{(6,7)}.

In this case, problem (P^1) with the step function costs on the links in set $\{M^{(0)}\}$ is constructed. This new constant cost routing solution is shown in Figure 4.

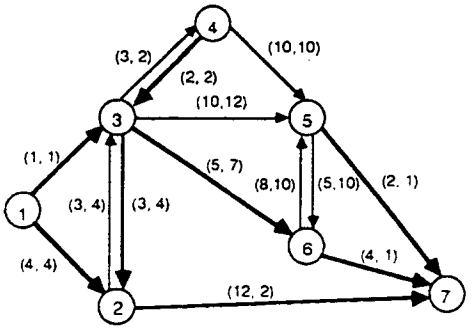


Fig. 4 Problem (P^1) and its nondominated paths.

The sets $\{Eff(B_i)^{(1)}\}$ of all nondominated paths from node i, i = 1, ..., 6, are:

{(1,2), (2,7)}, {(1,3), (3,6), (6,7)},
{(2,7)},
{(3,2), (2,7)}, {(3,6), (6,7)},
{(4,3), (3,2), (2,7)}, {(4,3), (3,6), (6,7)},
{(5,7)},
{(6,7)}.

Observe that the set $\{M^{(2)}\} = \varnothing$, and according to STEP 3 of the algorithm
$\{Eff^0(B_i)\} = VMIN \{ \{Eff(B_i)^{(1)}\} \cup \{Eff^0(B_i)^{(0)}\} \}$
for i = 1, ..., 6, is as follows:

{(1,2), (2,7)}, {(1,3), (3,6), (6,7)}, {(1,3), (3,5), (5,7)},
{(2,7)},
{(3,2), (2,7)}, {(3,6), (6,7)}, {(3,5), (5,7)},
{(4,5), (5,7)},
{(5,7)},
{(6,7)}.

In this case $\{Eff(B_i)\} = \{Eff^0(B_i)\}$.

V. CONCLUSIONS

This paper presents a method of approximation of the set of nondominated paths in a dynamic multiple criteria network with specially structured cost functions. The structure in the form of monotone increasing step functions makes the dynamic network very attractive for many applications.

The problem, being in the general class of time dependent multiple criteria path planning problems, can be consistently solved by two exact algorithms developed in [3]. In particular, Algorithm One utilizes a discrete time scale and finds the set of all nondominated paths form every node to the destination node in a dynamic multiple criteria network. Algorithm Two uses forward dynamic programming and finds the same set for a dynamic multiple criteria network with monotone increasing cost functions, after it has been restarted N-1 times (once from each node).

The solution procedure proposed in this paper is designed to make use of the special structure of the cost functions and involves solving a sequence of constant vector cost dynamic programming problems. No discrete time scale is required as in Algorithm One. It is not required to use a forward dynamic programming approach, restarting from each node, as in Algorithm Two.

The decrease in computation to obtain the set of nondominated paths may be quite significant. In our examples section, a seven node, thirteen link problem with two time dependent objective functions is solved and only two constant cost path planning problems are required. For comparison, Algorithm Two requires at least six routing problems in which the costs are computed during the iteration rather than in advance. Similarly, Algorithm One would require much more computational effort due to a time grid of at least 16 time steps. Thus the advantages of choosing the approximation method over Algorithm One or Algorithm Two for the step function costs with $\kappa << N$ are clearly seen.

REFERENCES

[1] R. Bellman, "On a routing problem," *Quarterly of Applied Mathematics*, vol. 16, pp. 87-90, 1958.

[2] M. M. Kostreva, M. M. Wiecek, and T. Getachew, "Optimization models in fire egress analysis for residential buildings," Fire Safety Science - Proceedings of the Third International Symposium, pp. 805-814, 1991.

[3] M. M. Kostreva, and M. M. Wiecek, "Time dependency in multiple objective dynamic programming," *Journal of Mathematical Analysis and Applications*, in press.

[4] M. M. Kostreva, and M. M. Wiecek, "Multicriteria decision making in fire egress analysis," Preprints of the IFAC/IFORS Workshop on Support Systems for Decision and Negotiation Processes, Poland, 1992.

[5] H. W. Corley, and I. D. Moon, "Shortest paths in networks with vector weights," *Journal of Optimization Theory and Applications*, vol. 46, pp. 79-86, 1985.

[6] R. Hartley, "Vector optimal routing by dynamic programming," in *Mathematics of Multiobjective Optimization*, P. Serafini, Ed. Vienna: Springer-Verlag, 1985, pp. 215-224.

[7] T. Ibaraki, "Enumerative approaches to combinatorial optimization, Part II," *Annals of Operations Research*, vol.11, pp. 343-440, 1987.

[8] I. E. Schochetman, and R. L. Smith, "Finite dimensional approximation in infinite dimensional mathematical programming", *Mathematical Programming*, vol. 54, pp. 307-333, 1992.

[9] T. Getachew, "A recursive algorithm for multi-objective network optimization with time-variant link-costs," Ph.D. Thesis, Management Science Program, Clemson University, 1992.

Appendix 5

"Transient Behavior in Multiple Criteria Path Planning Problems," Proceedings of 1992 IEEE International Conference on Systems, Man and Cybernetics, Volume 1, pp. 867-873, Chicago, Illinois, October 1992 (with T. Getachew).

Transient Behavior in Multiple Criteria Path Planning Problems

Teodros Getachew

Michael M. Kostreva
Department of Mathematical Sciences
Clemson University
Clemson, SC 29634-1907

Abstract **Multi-stage decision problems find a natural setting in the context of dynamic programming. An essential stipulation is that the underlying parameter space be time-invariant. There are numerous practical problems in multi-stage decision making, however, which involve parameters that are time-dependent; moreover, such dependency may exhibit non-monotonic or even discontinuous behavior. In such cases, classical Dynamic Programming is not applicable, as the Principle of Optimality may fail to hold.**

A recursive, constructive algorithm to solve multi-criteria problems with time dependent cost functions is presented. This method is a direct generalization of dynamic programming. Under these assumptions, the set of efficient decisions is itself a function of time. The algorithm is used to demonstrate an analysis of the transient behavior of this set by means of an example from fire egress modelling.

I. INTRODUCTION

Dynamic programming [1] is extended to the multi-objective, time independent case in [4]. The authors formally show that the Principle of Optimality can be used to find all the Pareto-Optimal paths from an origin node to the rest of the nodes in a directed network. In [2], it is noted that this problem is NP-hard, and an empirical investigation of the bicriterion label-correcting shortest path algorithm is carried out.

A theoretical contribution to the bicriterion shortest path problem is found in [4], where a two stage solution procedure is established. An early paper to consider the time-dependent shortest path problem is [3]. In this paper, the Principle of Optimality is extended in a straight-forward manner to the case where the transit times between nodes, a single criterion, are time-dependent and integral. Orda and Rom [10] consider the problem for different delay models. They also discuss the computational complexity of their proposed algorithms.

Philpott [11] presents a continous-time linear program formulation of the problem of finding the minimum-length path from an origin node to a destination node. Finally, in [8], Kostreva and Wiecek present two algorithms to solve the problem of finding all the non-dominated (Pareto-Optimal) paths from all nodes to a destination node. The first is a generalization of Cooke and Halsey [3] to the multi-objective case. The other generalizes the work of Kaufman and Smith [6] to the multi-objective context . Both of these algorithms inherit the restrictions of their single-objective analogues; the first, the integrality of transit times and expanded static network, and the second, the "no passing allowed" and monotonicity of costs.

The point of departure for the algorithm to be proposed in this paper is the observation that the approach of Cooke and Halsey, (which, because of the generality of the problems it is able to solve will be the one of interest here), is not a natural one, in that it attempts to solve the problem entirely by link augumentation; that is, by making the links the only fundamental repositories of information. This is a natural setting for the time-invariant case, where the link-cost pair is a well defined entity. In the time dependent setting however, it is evident that trying to preserve this well-definition is too costly, in terms of computational tractability.

The entity that enjoys well definition in the time variant case is the path-cost pair. However, searching for the solution entirely in the set of paths is equivalent to exhaustive path enumeration.

The algorithm presented in this paper is a combination of the two approaches. Fundamental to this algorithm is the notion of a partition of the path set; a partition into, a set of paths that are amenable to search via their links in a backward dynamic programming procedure, and the complementary set of paths whose costs are known through a forward 'costing' procedure. Starting from an initial partition consisting of the entire set of paths from a given node to the destination node, this procedure generates a unique sequence of partitions,.

Manuscript received August 1, 1992.

This research supported in part by Grant No. 60NANB0D1023, Building and Fire Research Laboratory, National Institute of Standards and Technology, Gaithersburg, MD

0-7803-0720-8/92 $3.00 ©1992 IEEE

recursively, each containing a better approximation to the solution set, until, after a finite number of iterations, the stopping criterion is attained, and the entire solution set determined. The formal elaboration of the algorithm follows.

II. FORMALISMS

Definition 1 : Let $\mathfrak{N}$, a set of n distinct points in $\mathfrak{R}^2$, called nodes, and $\mathfrak{L}$, a set of ordered pairs of distinct elements from $\mathfrak{N}$, called links, be given. A directed network with node set $\mathfrak{N}$ and link set $\mathfrak{L}$ is the set, $\mathcal{G} = \mathfrak{N} \cup \mathfrak{L}$.

Definition 2: An element (i,j) of $\mathfrak{L}$ is referred to as a directed link from node i to node j.

Definition 3 : A set $\mathcal{P}$ of elements of directed links is said to be a path from i to j, $i \neq j$, iff

$$\mathcal{P} = \{ (i, j_1), (j_1, j_2), \ldots, (j_{k-2}, j_{k-1}), (j_{k-1}, j)\}$$

$\mathcal{P}$ is said to have cardinality k.

Let a distinguished element d of $\mathfrak{N}$, called the destination node be given. Then $\mathbb{P}_i(\mathcal{G})$ shall denote the set of paths $\mathcal{P}$ with initial node i and final node d. The set of all such paths, emanating from nodes that are distinct from the destination node will be referred to simply as paths. This set is defined by the equation:

$$\mathbb{P}(\mathcal{G}) = \bigcup_{i \in \mathfrak{N} \setminus \{d\}} (\mathbb{P}_i(\mathcal{G})). \tag{1}$$

Notation 1: $\mathbb{P}_i^{(k)}(\mathcal{G})$ shall denote the set of paths in $\mathbb{P}_i(\mathcal{G})$ with each path of cardinality at most k.

Notation 2: $\mathbb{P}_i^{k}(\mathcal{G})$ shall denote the set of paths in $\mathbb{P}_i(\mathcal{G})$ with each element of cardinality exactly k.

Definition 4: A subset $\mathcal{P}'$ of a path $\mathcal{P} \in \mathbb{P}(\mathcal{G})$ said to be a subpath of $\mathcal{P}$ if and only if $\mathcal{P}' \in \mathbb{P}(\mathcal{G})$.

Definition 5: Let $\mathfrak{F}$ be a set of functions with domain $\mathfrak{R}^+ \cup \{0\}$ and range $\mathfrak{R}^+$, bounded above and below on bounded intervals.

The link-transition cost functions are then given by the range of the function $\mathfrak{T}$ where:
$\mathfrak{T} : \mathfrak{L} \rightarrow \mathfrak{F}$, given by

$$\mathfrak{T}((i,j)) = \mathfrak{T}_{ij}(t) \in \mathfrak{F}. \tag{2}$$

The cost on a link, as noted above is a function, not only of time, but also of the paths to which it belongs. Given a particular path, and a particular link on it, the cost on the link is determined by the time of arrival, via the path, at the initial node of the link. This is formalized in terms of the arrival function.

Definition 6: Let $\mathcal{P} = \{(i, j_1), (j_1, j_2), \ldots, (j_{k-2}, j_{k-1}), (j_{k-1}, d)\}$ be a path in $\mathbb{P}_i^k(\mathcal{G})$. The arrival function $\mathcal{A}: \mathbb{P}(\mathcal{G}) \times \mathfrak{N} \rightarrow \mathfrak{R}^+$ is defined recursively on initial link-nodes as follows:

$$\mathcal{A}(\mathcal{P}, i) = t^o; \tag{3}$$

Suppose $\mathcal{A}(\mathcal{P}, j_{k-1})$ has been defined for $r \leq s-1$; let $(j_s, j_{s+1}) \in \mathcal{P}$. Then,

$$\mathcal{A}(\mathcal{P}, j_{s+1}) = \mathcal{A}(\mathcal{P}, j_s) + \mathfrak{T}_{j_s j_{s+1}}(\mathcal{A}(\mathcal{P}, j_s)). \tag{4}$$

Definition 7: $C: \mathfrak{L} \rightarrow \mathfrak{T} \times \mathfrak{T} \times \ldots \times \mathfrak{T}$ such that,

$$C(i,j) = (c_{ij}^{(1)}(t), \ldots, c_{ij}^{(p)}(t)), \tag{5}$$

where each $c_{ij}^{(k)} \in \mathfrak{F}$. The "path costing" functions Γ_o and Γ are defined next.

Definition 8: Let

$$c_o^{(k)}(n_i, n_j) = \inf_{t \in I(n_i, n_j)} c_{n_i n_j}^{(k)}(t) \tag{6}$$

where,

$$I(n_i, n_j) = [\alpha, \beta], \tag{7}$$

with

$\alpha = \min \mathcal{A}(\mathcal{P}, n_i)$ and $\beta = \max \mathcal{A}(\mathcal{P}, n_j)$, $(n_i, n_j) \in \mathcal{P}$.

Now, let $\mathcal{P} \in \mathbb{P}_i(\mathcal{G})$ be of cardinality k. Then

$\Gamma_o: \mathbb{P}(\mathcal{G}) \rightarrow \mathfrak{R}^+ \times \mathfrak{R}^+ \times \ldots \times \mathfrak{R}^+$ is defined by

$$\Gamma_o(\mathcal{P}) = \vec{c}_o(i, j_1) + \vec{c}_o(j_{k-1}, d) +$$

$$\sum_{s=1}^{k-2} (\vec{c}_o\ (j_s, j_{s+1})), \qquad (8)$$

where

$$\vec{c}_o\ (j_q, j_s) = (c_o^{(1)}(j_q, j_s), \ldots, c_o^{(k)}(j_q, j_s)). \qquad (9)$$

Definition 9: Forward costing is defined as follows. $\Gamma: \mathbb{P}(\mathcal{G}) \to \mathcal{R}^+ \times \mathcal{R}^+ \times \ldots \times \mathcal{R}^+$ is defined by,

$$\Gamma(\mathcal{P}) = \vec{c}_{ij_1}(t^o) + \vec{c}_{j_{k-1}d}(\mathcal{Q}(\mathcal{P}, j_{k-1})) + \sum_{s=1}^{k-2} (\vec{c}_{j_s j_{s+1}}(\mathcal{Q}(\mathcal{P}, j_s))) . \qquad (10)$$

Finally, the notion of efficiency is formally stated in terms of the notation given above.

Definition 10: Let $\mathcal{P} \in \mathbb{P}^{(k)}(\mathcal{G})$. Then,

$\mathcal{P} \in \mathcal{E}ff^{(k)}_i(P_o)$ if and only if the set,

$\{\mathcal{P}': \mathcal{P}' \in \mathbb{P}^{(k)}(\mathcal{G})$ and $\Gamma_o(\mathcal{P}') \leq \Gamma_o(\mathcal{P})$ and $\Gamma_o(\mathcal{P}') \neq \Gamma_o(\mathcal{P})\}$ is empty.

Similarly, $\mathcal{P} \in \mathcal{E}ff^{(k)}_i(P)$ if and only if the set, $\{\mathcal{P}': \mathcal{P}' \in \mathbb{P}^{(k)}(\mathcal{G})$ and $\Gamma(\mathcal{P}') \leq \Gamma(\mathcal{P})$ and $\Gamma(\mathcal{P}') \neq \Gamma(\mathcal{P})\}$ is empty.

III. ALGORITHM

The algorithm that is presented below has the following features:

1. It is a recursive algorithm in which only the information generated in a previous iteration is used in the execution of a subsequent one;
2. It takes place both in the set of links, in a backward dynamic programming phase, and the set of paths, in a forward integration phase.

A. Iteration I_o:

Given a network, with each link having an associated transition cost function as defined above, the algorithm begins with the initial partition of the set of paths; namely, the partition consisting of the empty set and the entire set of paths with all links costed as in Γ_o. Note that this costing yields a unique network, as each link transition cost function has a unique infimum. What follows then is an application of backward dynamic programming to this network. Formally we have,

$$F_{0i}^{(1)} = \text{VMIN}\ \{\Gamma_o(\mathcal{P}) : \mathcal{P} \in \mathbb{P}_i^1(\mathcal{G})\}; \qquad (11)$$

$$F_{0i}^{(2)} = \text{VMIN}\ \{\Gamma_o(\mathcal{P}): \Gamma_o(\mathcal{P}) = \vec{c}_o(i,j) + f_{0j}^{(1)}, f_{0j}^{(1)} \in F_{0j}^{(1)}\}; \qquad (12)$$

$$F_{0i}^{(r)} = \text{VMIN}\ \{\Gamma_o(\mathcal{P}): \Gamma_o(\mathcal{P}) = \vec{c}_o(i,j) + f_{0j}^{(r-1)}, f_{0j}^{(r-1)} \in F_{0j}^{(r-1)}\}; \qquad (13)$$

...

$$F_{0i}^{(n-1)} = \text{VMIN}\ \{\Gamma_o(\mathcal{P}): \Gamma_o(\mathcal{P}) = \vec{c}_o(i,j) + f_{0j}^{(n-2)}, f_{0j}^{(n-2)} \in F_{0j}^{(n-2)}\}; \qquad (14)$$

The backward dynamic programming stage terminates with the set $F_{0i}^{(n-1)}$.

Let,

$$F_{0i} \equiv F_{0i}^{(n-1)},$$

and

$$\mathbb{P}(F_{0i}) = \{\mathcal{P} : \mathcal{P} \in \mathbb{P}(\mathcal{G}) \text{ and } \Gamma_o(\mathcal{P}) \in F_{0i}\}. \qquad (15)$$

The true cost of each path whose cost is in the set F_{0i} is now found by direct evaluation, as a function of start-time t^o of journey on this path, via the costing function Γ. If this cost differs from its cost under Γ_o, the path is added to the set V_{0i}. Let

$$V_{0i} = \{\mathcal{P}:\ \mathcal{P} \in \mathbb{P}(F_{0i}) \text{ and } \Gamma_0(\mathcal{P}) \leq \Gamma(\mathcal{P}) \text{ and } \Gamma_0(\mathcal{P}) \neq \Gamma(\mathcal{P})\}. \quad (16)$$

If V_{0i} is empty, then STOP; $\mathbb{P}(F_{0i}) = \mathcal{C}ff_i(P)$;

else let,

$$\mathcal{U}_{0i} = \emptyset \quad (17)$$

and

$$\mathcal{U}_{1i} = V_{0i} \cup \mathcal{U}_{0i}. \quad (18)$$

Suppose now that I_{r-1} and $\mathcal{U}_r$ have been defined. Then, define iteration I_r via the set S_r.

B. Iteration I_r

Definition 11: The set S_r of forbidden paths is defined by the equation:

$$S_r = \{\mathcal{P}:\ \mathcal{P} \in \mathbb{P}(\mathcal{G}) \text{ such that } \mathcal{P} \text{ is a subpath of } \mathcal{P}',\ \mathcal{P}' \in \mathcal{U}_r\}.$$

This set defines the partition and the costing for iteration I_r. The path-costing function for the rth iteration is now defined by:

Definition 12: $\Gamma_r: \mathbb{P}(\mathcal{G}) \rightarrow \mathcal{R}^+ \times \mathcal{R}^+ \times \ldots \times \mathcal{R}^+$, where,

$$\Gamma_r(\mathcal{P}) = \Gamma_0(\mathcal{P}) \text{ if } \mathcal{P} \notin S_r, \quad (19)$$

$$= \Gamma(\mathcal{P}) \text{ otherwise.} \quad (20)$$

Let

$$S_{ri} \equiv \mathbb{P}_i(\mathcal{G}) \cap S_r,$$

and

$$S^k_{ri} \equiv \mathbb{P}^k_i(\mathcal{G}) \cap S_r.$$

The backward dynamic programming is applied to the set of paths that are in the complement of the set S_{ri}.

$$F^{(1)}_{ri} = \text{VMIN}\{\Gamma_r(\mathcal{P}) : \mathcal{P} \in \mathbb{P}^1_i(\mathcal{G}), \text{ and } \mathcal{P} \notin S_{ri}\}; \quad (21)$$

Paths of length at least two in the complement of S_{ri} can take two forms: those paths all of whose links but the latest come from the previous vector-minimization, and those all of whose links except the latest come from a set S_{ri}. Accordingly, vector-minimization over sets of paths of cardinality greater than one must involve, not only paths from the preceding vector-minimization step, but also paths from appropriate forbidden sets.

$$F^{(2)}_{ri} = \text{VMIN}\{\{\Gamma_r(\mathcal{P}):\ \Gamma_r(\mathcal{P}) = f^{(1)}_{rj} + \vec{c}_0(i,j),\ f^{(1)}_{rj} \in F^{(1)}_{rj},\ \mathcal{P} \notin S_{ri}\} \cup \{\Gamma_r(\mathcal{P}) :\ \Gamma_r(\mathcal{P}) = \Gamma_0(\mathcal{P}') + \vec{c}_0(i,j),\ \mathcal{P}' \in S^1_{rj} \text{ and } \mathcal{P} \notin S_{ri}\}\}. \quad (22)$$

...

The backward dynamic programming ends with the evaluation of the set $F^{(n-1)}_{ri}$.

$$F^{(n-1)}_{ri} = \text{VMIN}\{\{\Gamma_r(\mathcal{P}):\ \Gamma_r(\mathcal{P}) = f^{(n-2)}_{rj} + \vec{c}_0(i,j),\ f^{(n-2)}_{rj} \in F^{(n-2)}_{rj},\ \mathcal{P} \notin S_{ri}\} \cup \{\Gamma_r(\mathcal{P}) :\ \Gamma_r(\mathcal{P}) = \Gamma_0(\mathcal{P}') + \vec{c}_0(i,j),\ \mathcal{P}' \in S^{n-2}_{rj} \text{ and } \mathcal{P} \notin S_{ri}\}\}. \quad (23)$$

Let

$$F_{ri} = \text{VMIN}\{F^{(n-1)}_{ri} \cup \{S_{ri}\}\} \quad (24)$$

where

$$\{S_{ri}\} \equiv \{\Gamma_r(\mathcal{P}):\ \mathcal{P} \in S_{ri}\}.$$

Letting,

$$\mathbb{P}(F_{ri}) \equiv \{\mathcal{P}:\ \mathcal{P} \in \mathbb{P}_i(\mathcal{G}) \text{ and } \Gamma_r(\mathcal{P}) \in F_{ri}\},$$

the set V_{ri} is defined to be the set of paths whose cost under the costing Γ_r differs from that under Γ.

Definition 13: $V_{ri} \equiv \{\mathcal{P}: \ \mathcal{P} \in \mathbb{P}(F_{ri})$ and $\Gamma_r(\mathcal{P}) \leq \Gamma(\mathcal{P})$ and $\Gamma_r(\mathcal{P}) \neq \Gamma(\mathcal{P})\}$.

If V_{ri} is empty, STOP, $\mathbb{P}(F_{ri}) = \mathcal{E}ff_i(P)$; else let,

$$\mathcal{U}_{(r+1)i} \equiv V_{ri} \cup \mathcal{U}_{ri}\ ; \tag{25}$$

$$\mathcal{U}_{r+1} \equiv \bigcup_{i \neq d} \mathcal{U}_{(r+1)i}. \tag{26}$$

IV. THEOREMS

Note: In the sequel, the symbol $\leq\neq$ shall stand for "at most as large as, vector-wise, but distinct from".The following theorems have been established (see [5] for proofs).

Theorem 1: The algorithm terminates after a finite number of iterations.

Let

$$\mathbb{P}(\mathcal{G}_r') \equiv \mathbb{P}(\mathcal{G}) \backslash S_r,$$

and

$$\mathbb{P}_i(\mathcal{G}_r') \equiv \mathbb{P}(\mathcal{G}_r') \cap \mathbb{P}_i(\mathcal{G})$$

Define $\mathcal{E}ff(\mathbb{P}_i(\mathcal{G}_r'))$ by the condition:

$\mathcal{P} \in \mathcal{E}ff(\mathbb{P}_i(\mathcal{G}_r'))$ if and only if the set

$$\{\mathcal{P}': \mathcal{P}' \in \mathbb{P}_i(\mathcal{G}_r') \text{ and } \Gamma_0(\mathcal{P}') \leq\neq \Gamma_0(\mathcal{P})\}$$

is empty.

Theorem 2: $\mathbb{P}(F_{ri}^{n-1}) = \mathcal{E}ff(\mathbb{P}_i(\mathcal{G}_r'))$. (27)

Define the set $\mathcal{E}ff_i(P_r)$ by:

$\mathcal{P} \in \mathcal{E}ff_i(P_r)$ if and only if the set

$\{\mathcal{P}': \mathcal{P}' \in \mathbb{P}_i(\mathcal{G})$ and $\Gamma_r(\mathcal{P}') \leq\neq \Gamma_r(\mathcal{P})\}$ is empty.

Theorem 3: $\mathbb{P}(F_{ri}) = \mathcal{E}ff_i(P_r)$. (28)

In the next theorem, the main theoretical result of this paper is stated; namely that after a finite number of iterations, the algorithm determines the entire set of efficient paths, for all possible initial nodes.

Theorem 4: Let the algorithm terminate after t iterations. Then,

$$\mathbb{P}(F_{ti}) = \mathcal{E}ff_i(P). \tag{29}$$

V. APPLICATION

Increasing interest is being shown in recent years in fire egress analysis [7]. Our application shows how the algorithm can be used to generate the data for the analysis of the transience of efficient paths arising in this important problem.

Consider the following problem. A fire starts in a room in a residential building. The occupants of the building, in attempting to make their way to safety, will continuously need to make trade-off decisions with respect to possibly conflicting criteria. These decisions are made against a background of a continuously evolving environment, governed by the fire, and their interactions with it. A question of interest is the determination of optimal egress paths from all rooms to a place of safety. The abstract setting for this problem is a network with nodes representing rooms, and links representing passageways. Each link has associated with it a vector of time-dependent costs, representing the various time-variant criteria. The problem of interest then becomes finding the set of all non-dominated paths from each node (room) to a destination node. This problem being the original motivation for the development of the algorithm, it is solved by a direct application of it.

Consider the typical family home below, (fig. 1). A fire starts at time t = 0, in the kitchen, and spreads through the house, impacting the passability of corridors, and the physiological conditions of the occupants, according to specific functions of time. The occupants have two aims: a) to maximize their stay in the house after the fire starts, in order to give aid to other occupants, to save their

belongings, etc. and b) to minimize the risk to their lives posed by the smoke, the heat and the toxicants created by the fire. Note that the objective, consisting of two aims that are in contradiction to one another, requires a multiple objective formulation to be adequately modelled.

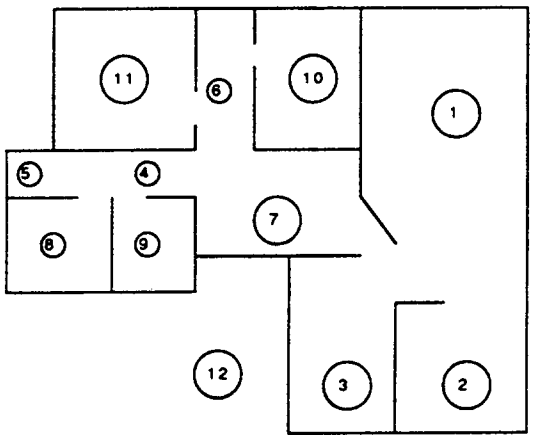


Fig. 1. The floor plan of a family home.

Key

1 = Living Room	2 = Dining Room
3 = Kitchen	4 = Corridor
5 = Corridor	6 = Corridor
7 = Foyer	8 = Children's Bedroom
9 = Children's Bedroom	10 = Guest Bedroom
11 = Master Bedroom	12 = Safety

This example will attempt to consider how the following question may be addressed: What is the transient behavior of the set of efficient paths ? There are three types of time-variant costs that pertain to a link; they are,

1. Transit time: the time it takes to get from the initial node to the final node of the link (i,j) ; the notation for this is $T_{(i,j)}(t)$,
2. Time left till impassability: the time until the link (i,j) becomes impassable; the notation for this is $L_{(i,j)}(t)$,
3. Risk: the amount of risk to human survival that the particular link (i, j) carries; the notation for this is $R_{(i,j)}(t)$. See fig. 2.

Transit time across a link is depicted as a function of arrival time at the initial node of the link, and its final node. It is a non-decreasing function, in keeping with the nondecreasing difficulty of movement with the progress of the fire through the house. Time to impassibility is also a function of the arrival time at the initial node of the link, and its final node. It is a non-increasing function, since the impassibility of a node increases with time.

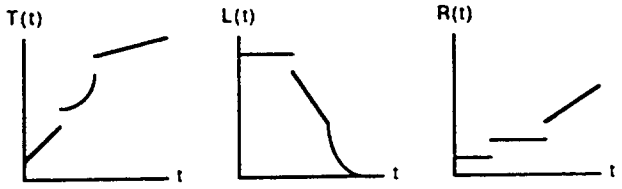


Fig. 2. Instances of the functions T, L and R.

The third function, risk to human well-being, is modelled as a non-decreasing function of arrival time at the initial node of a link, and its final node.

In the implementation of the application, these functions, were approximated by linear interpolation. See [5] for details.

VI. RESULTS

Using the data given in [5], the following paths have been found to be non-dominated egress paths for at least one exit start time from among: t^o = 0, 10, 20 , 30, 40, 50, 60, 70, 80, 90, 150, 300. Table I will be used to identify the various paths.

The set of efficient paths from a given room to safety is a function of exit start time. It is of interest to examine the transient behavior of such a set. Table II depicts this behavior for the set of paths listed in Table I. An x indicates that the given path is efficient for the indicated start time, while an o indicates otherwise.

VII. DISCUSSION

As is evident from the computational results depicted in Table II, the Pareto set is a function of the exit start-time t^o. For the set of functions $R_{(i,j)}(t)$, $L_{(i,j)}(t)$ and $T_{(i,j)}(t)$ in this problem, all implicitly dependent on the evolution of the fire-dictated environment, it is apparent that this Pareto set reaches equilibrium as $t^o \rightarrow \infty$. It is of interest to know the amount of time required to do so in general.

It is noteworthy that transit time is is not included in the set of objectives to which the vector minimization is applied. This feature of the algorithm may be exploited to introduce objectives which may be functions of different measures of time; see [5], application two..

From the computation, one may observe that certain rooms are only visited by stable, Pareto paths, while others have Pareto paths which appear, disappear and appear again. This feature may be used to evaluate the design of a given building. A more complete analysis would however require

consideration of the problem of different exit start-times for different occupants. This would generalize the results of this paper, obtained under the condition of simultaneous exit start-times for all occupants, and is left to future research.

TABLE I.
KEY FOR THE SET OF PARETO PATHS

Origin Node	Path	Symbol
1	1-2-3-5-7-12	[1,1]
	1-2-3-5-8-4-7-12	[1,2]
	1-5-7-12	[1,3]
	1-5-8-4-7-12	[1,4]
2	2-1-5-7-12	[2,1]
	2-1-5-8-4-7-12	[2,2]
	2-3-5-7-12	[2,3]
	2-3-5-8-4-7-12	[2,4]
3	3-4-7-12	[3,1]
	3-2-1-5-7-12	[3,2]
	3-5-7-12	[3,3]
	3-5-8-4-7-12	[3,4]
	3-2-1-5-8-4-7-12	[3,5]
4	4-7-12	[4,1]
5	5-7-12	[5,1]
	5-8-4-7-12	[5,2]
6	6-5-7-12	[6,1]
	6-5-8-4-7-12	[6,2]
	6-10-5-7-12	[6,3]
	6-10-5-8-4-7-12	[6,4]
7	7-12	[7,1]
8	8-5-7-12	[8,1]
9	9-5-7-12	[9,1]
	9-5-8-4-7-12	[9,2]
10	10-5-7-12	[10,1]
	10-5-8-4-7-12	[10,2]
	10-6-5-7-12	[10,3]
	10-6-5-8-4-7-12	[10,4]
11	11-6-5-7-12	[11,1]
	11-6-10-5-7-12	[11,2]
	11-6-5-8-4-7-12	[11,3]
	11-6-10-5-8-4-7-12	[11,4]

TABLE II.
TRANSIENT BEHAVIOR OF THE SET OF PARETO PATHS
FOR THE GIVEN SET OF START-TIMES

Time/ Path	0	10	20	30	40	50	60	70	80	90	150	300
[1,1]	o	x	x	x	o	o	o	x	x	x	x	x
[1,2]	o	x	x	x	o	o	o	x	x	x	x	x
[1,3]	x	x	x	x	x	x	x	x	x	x	x	x
[1,4]	x	o	o	x	x	x	x	o	o	o	o	o
[2,1]	o	x	x	x	x	x	x	x	x	x	x	x
[2,2]	o	x	x	x	x	x	x	x	x	x	x	x
[2,3]	x	o	o	o	o	o	o	o	o	o	o	o
[2,4]	x	o	o	o	o	o	o	o	o	o	o	o
[3,1]	x	x	x	x	x	x	x	x	x	x	x	x
[3,2]	o	o	x	x	x	x	x	x	x	x	x	x
[3,3]	o	o	x	x	o	o	o	o	o	o	o	o
[3,4]	x	x	x	o	x	x	x	x	x	x	x	x
[3,5]	x	x	o	o	o	o	o	o	o	o	o	o
[4,1]	x	x	x	x	x	x	x	x	x	x	x	x
[5,1]	x	x	x	x	x	x	x	x	x	x	x	x
[5,2]	x	x	x	x	x	x	x	x	x	x	x	x
[6,1]	x	x	x	x	x	x	x	x	x	x	x	x
[6,2]	x	o	x	x	x	x	x	x	x	x	x	o
[6,3]	x	x	x	x	x	x	x	x	x	x	x	x
[6,4]	x	x	x	x	x	x	x	x	x	x	x	x
[7,1]	x	x	x	x	x	x	x	x	x	x	x	x
[8,1]	x	x	x	x	x	x	x	x	x	x	x	x
[9,1]	x	x	x	x	x	x	x	x	x	x	x	x
[9,2]	x	x	x	x	x	x	x	x	x	x	x	x
[10,1]	x	x	x	x	x	x	x	x	x	x	x	x
[10,2]	x	x	x	x	x	x	x	x	x	x	x	x
[10,3]	x	o	o	o	o	o	o	o	o	o	o	o
[10,4]	x	o	o	o	o	o	o	o	o	o	o	o
[11,1]	x	x	x	x	x	x	x	x	x	x	x	x
[11,2]	x	x	x	x	x	x	x	x	x	x	x	x
[11,3]	o	x	x	x	x	x	x	x	x	x	o	o
[11,4]	x	x	x	x	x	x	x	x	x	x	x	x

REFERENCES

[1] R. Bellman, On a routing problem, Quarterly of Applied Mathematics 16 (1958), 87-90.

[2] J. Brumbaugh-Smith and D.Shier, An empirical investigation of some bicriterion shortest path algorithms, European Journal of Operational Research 43 (1989) 216-224.

[3] K. L. Cooke and E. Halsey, The shortest route through a network with time-dependent internodal transit times, Journal of Mathematical Analysis and Applications 14 (1966), 493-498.

[4] H. W. Corley and I. D. Moon, Shortest paths in networks with vector weights, Journal of Optimization Theory and Applications 46 (1985), 79-86.

[5] T. Getachew, An algorithm for multiple-objective network optimization with time variant link-costs, Ph.D. Thesis, Management Science, Clemson University, 1992.

[6] D. E. Kaufman and R. L. Smith, Minimum travel time paths in dynamic networks with application to intelligent vehicle-highway systems, University of Michigan, Transportation Research Institute, IVHS Technical Report-90-11, 1990.

[7] M. M. Kostreva, M. M. Wiecek and T. Getachew, Optimization models in fire egress analysis for residential buildings, Proceedings of the Third International Symposium on Fire Safety Science, Edinburgh, United Kingdom, July 8-12, 1991, 805-814.

[8] M. M. Kostreva and M. M. Wiecek, Time dependency in multiple-objective dynamic programming, Department of Mathematical Sciences, Clemson University, Technical Report #601, 1991, Journal of Mathematical Analysis and Applications, in press.

[9] J. Mote, Ishwar Murthy and David L. Olson, A parametric approach to solving bicriterion shortest path problems, European Journal of Operational Research, 53 (1991) 81-92.

[10] A. Orda and R. Rom, Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length, Journal of the Association for Computing Machinery 37 (3) (1990), 607-625.

[11] A. B. Philpott, A class of continous-time shortest-path problems, SIAM Journal on Control and Optimization, in press.

Appendix 6
"Time Dependency in Multiple Objective Dynamic Programming," Journal of Mathematical Analysis and Applications 173, 289-307 (1993) (with M. Wiecek).

Reprinted from JOURNAL OF MATHEMATICAL ANALYSIS AND APPLICATIONS
Vol. 173, No. 1, February 1993
Printed in Belgium

Time Dependency in Multiple Objective Dynamic Programming

MICHAEL M. KOSTREVA AND MALGORZATA M. WIECEK

Department of Mathematical Sciences, Clemson University, Clemson, South Carolina 29634-1907

Submitted by E. Stanley Lee

Received May 1, 1991

1. INTRODUCTION AND PRELIMINARIES

The problem of planning paths in a network structure is important for many applications. Interest in path planning is strong in transportation, telecommunications, computer design, and fire hazard analysis. Such a problem is sometimes called a routing problem, or a shortest path problem. One of the earliest solutions to the problem was given by Bellman [1]. Under the assumptions of constant travel times on each link, dynamic programming was applied to compute the path of minimum travel time through the network, from any node to a given destination node. Bellman applied the functional equations approach to devise an iterative algorithm which converges to the solution in at most $N-1$ steps for a network with N nodes.

Interest in the problem arose recently from fire hazard analysis [12]. In particular, it is desired to construct realistic models of the egress of humans from a residential building which is involved in fire. To add realism to the models, it was suggested to consider dynamic networks (costs on links are functions of time) and multi-objective behavior of humans. It seems clear that a human, taking a quick look at a scene in a burning building, will not consider only the time required to travel along a path, but also whether the path is cluttered with obstructions, whether smoke is present, whether fire is present, the distance of the path, the sound of someone calling, and so on. Human integration and automatic consideration of trade-offs are evident in decisions that they make. Although it is not evident that the complete data needed to solve a complex model such as will be proposed is actually available to humans in egress situations, the models are viewed as idealizations to use for standards or benchmarks. Once the model solutions

0022-247X/93 $5.00

are known, it is unlikely that humans will be able to do real-time path planning which is as good as these solutions. Hence, they are considered as lower bounds on what can be expected in real egress scenarios. If these lower bound or ideal values represent unsatisfactory outcomes, then decision makers should be made aware of the potential hazards and take appropriate remedial actions.

Work on dynamic programming for path planning in networks with dynamic cost functions has its roots in the paper of Cooke and Halsey [5]. Travel times on links were considered as general functions of time, and a grid of discrete values of time was superimposed. The functions were to be evaluated at the arrival time at a node. This type of evaluation is now known as the "frozen link" model of cost evaluation for dynamic networks [14]. This dynamic cost approach considers only travel times, and indeed, cannot comprehend other types of costs such as distances, etc. It will be shown in this paper how to modify the dynamic programming approach to include other types of cost functions in a multiple objective dynamic programming approach to path planning in networks.

Other related research on dynamic programming considering time dependent parameters includes Sebastian [16] and Li and Haimes [13] and the recent analysis of Orda and Rom [14]. Sebastian and Li and Haimes are concerned with discrete dynamical systems and their control under time varying constraints and parameters. The state equations they consider are replaced by the network structure in path planning. Hence, the results do not transfer easily to path planning in networks. Orda and Rom [14] are more concerned with computational complexity of path planning and other alternative algorithms under time varying costs. Their focus is restricted to the construction of a single path from one origin to a single destination. Hence the complexity results they obtain must be scaled up accordingly when computing all paths under the time varying assumption. Two separate objectives are considered, time and distance, but only one of these objective functions is present in any one model. Bertsekas and Gallager [2] ask a question (in the exercises) about how to handle a path planning problem in which one cost on one link increases at a given time. They suggest that there is a simple modification to an existing algorithm to handle such a situation. Very recent work by Kaufman and Smith [11] and Evans *et al.* [8] suggest that there are economical computational strategies available in the case of specially structured dynamic programming problems. For path planning, there are similar observations about structure and how to compute more effectively. Algorithms to effectively compute time dependent path planning solutions with multiple objectives will be introduced in this paper, under some assumptions about the time varying cost structure which are motivated by the applications. These assumptions are similar to those make by Kaufman and Smith for

the single objective case. It is interesting to note that these assumptions arose independently and simultaneously from two separate applications, transportation planning in congested road networks and fire hazard analysis. It is quite likely that the assumptions are widely applicable in many diverse settings.

A new development in dynamic programming theory was presented by Ibaraki [10], who introduced the concept of congener relaxation and related technique that he called successive sublimation dynamic programming. The original dynamic programming problem is relaxed and subsequent sublimations of the relaxation are examined until one of them is congener to the original problem, which guarantees availability of its solution. Relationships of this technique to our results are yet unknown.

Multiple objective dynamic programming developed concurrently to dynamic programming with time varying costs, but more research exists on the latter topic. Initially, Brown and Strauch [3] considered multiple objective functions with a latticial order. Daellenbach and De Kluyver [7], seemed to be unaware of Brown and Strauch, and they showed how to compute the set of nondominated paths in a network under the ordering of the nonnegative orthant. The theory behind the computation was not included in paper [7]. Multiple objective analysis of discrete dynamical systems is considered by Perevozchikov [15], while papers [6, 9] consider vector routing problems in networks with constant costs on the links. Carraway and Morin [4] consider a generalization of dynamic programming, but their work does not handle the time dependent cost structure of interest. Finally, Li and Haimes [13] consider the discrete dynamical systems control problem with multiple objectives and time dependent constraints. Again, the network structure precludes such work from application to the path planning problem of this paper.

The above discussion of progress to date in dynamic programming with time varying cost structure and multiple objective functions demonstrates that many related papers exist, but that evidently none will handle the object of interest: path planning in networks which comprehends multiple time varying objective functions. From this brief introduction to the problem and the existing literature, we now proceed to the formal framework for our research.

The mathematical framework we consider is a general network, not assumed to be acyclic. It consists of a set of nodes $\{1, 2, ..., N\}$ and a set of links which indicate connections between nodes, i.e., $\{(i_1, i_2), (i_3, i_4), ...\}$. The links' directions are indicated by the order of the indices. So, (3, 4) is the link from node 3 to node 4, while (4, 3) is the link from node 4 to node 3. A path from node i_0 to node i_p is a sequence of links $P = \{(i_0, i_1), (i_1, i_2), ..., (i_{p-1}, i_p)\}$ in which the initial node of each arc is the same as the terminal node of the preceding arc in the sequence and $i_0, ..., i_p$

are all distinct nodes. Let Π be the set of all feasible paths in the network which have the form

$$\{(i_1, i_2), (i_2, i_3), (i_3, i_4), ..., (i_{S-1}, i_S)\}, \qquad \text{where} \quad 1 \leqslant i_1, i_S \leqslant N.$$

Each link carries one or more attributes (i.e., time to travel, distance to travel, etc.) which we think of as cost functions. The cost (vector) of a link (i, j) applies to all paths which include link (i, j). The cost functions $(c_{ij}: R^+ \to R^{m+})$ are assumed to be positive vector valued functions of time, and are not assumed to be continuous. Let $[c_{ij}(t)]_1$ be the time to travel from node i to node j, given that travel starts at time t. The cost to traverse a path p in Π is defined to be

$$[c(p)] = \sum_{(i,j) \in p} [c_{ij}(t)].$$

A path in Π is a nondominated path if there is no other path p' in Π with $[c(p')] \leqslant [c(p)]$ and $[c(p')]_r < [c(p)]_r$ for some $r \in \{1, ..., m\}$, where symbol $\leqslant$ in the vector inequality denotes $[c(p')]_r \leqslant [c(p)]_r$ for $r = 1, ..., m$.

The organization of this paper is as follows. Section 2 includes all the theoretical results developed in the paper. In two subsections, two different approaches and algorithms to solving time dependent multiple criteria routing problems are presented. The first applies backward dynamic programming and solves the routing problem generating all nondominated paths leading from every node in the network to the destination node. The second approach solves the routing problem for the set of feasible paths which lead from the origin node to all other nodes in the network. The analysis shows how adoption of forward dynamic programming leads to a new version of the principle of optimality that can deal with a general class of dynamic multiple objective networks. The relationship between the forward and backward case is also explored. Examples that apply the two algorithms are included in Section 3.

2. Multiple Criteria Time Dependent Dynamic Programming Algorithms

2.1. *Backward Dynamic Programming Case*

In this section we present an algorithm for computing the set of all nondominated paths in the network, as introduced in the previous section. The algorithm is based on Bellman's principle of optimality [1]. We generalize the approach to finding the shortest (fastest) route through a network developed by Cooke and Halsey [5] to the multiple criteria case.

Assume a discrete time scale $S_T = \{t_0, t_0+1, t_0+2, ..., t_0+T\}$, $t_0 > 0$. Accordingly, assume that all the functions $[c_{ij}(t)]_k$, $i, j = 1, 2, ..., N$, $i \neq j$, $k = 1, 2, ..., m$, have positive values for $t \in S_T$. $[c_{ij}(t)]_1$ is the travel time from node i to node j, given that the arrival time at node i is t and it is assumed to be a positive integer. The number T is taken to be an upper bound on the total travel time required to go from any node in the network to node N. For example, $\max_{1 \leq i \leq N}\{[c_{iN}(t_0)]_1\}$ is an upper bound. If the nodes are not all connected to node N, then the number T is only implicitly defined. Since there are only a finite number of paths with finite cost vectors which are feasible, the value of T exists.

Now we introduce the following sets that are defined for $t \in S_T$ and $i = 1, 2, ..., N-1$:

$\{E_i(t)\}$—the set of all paths in the network which leave node i at time $t \in S_T$ and reach node N;

$\{\text{Eff}(E_i(t))\}$—the set of all nondominated paths which leave node i at time $t \in S_T$ and reach node N;

$\{[F_i(t)]\}$—the set of vector costs of all nondominated paths in $\{\text{Eff}(E_i(t))\}$;

$\{E_i(t)^{(k)}\}$—the set of all paths of at most k links, leaving node i at time t and reaching node N at or before time $t_0 + T$, $k = 2, 3, ...$;

$\{[F_i(t)^{(k)}]\}$—the set of vector costs of all nondominated paths in $\{E_i(t)^{(k+1)}\}$. Obviously, $\{\text{Eff}(E_i(t))\} \subseteq \{E_i(t)\}$.

According to the frozen link model, we also assume that upon the arrival time at node i the vector cost of link (i, j) for all j such that (i, j) exists, is an easily computed constant. Thus, the arrival time at node j is $t + [c_{ij}(t)]_1$.

Let $[\infty]$ and $\{\infty\}$ denote the vector and the set of m-component vectors such that each component is equal to infinity, respectively. Let $\{0\}$ denote a set containing the zero vector.

Bellman's principle of optimatlity [1] has been applied to multiple objective dynamic programming [3, 7, 9]. For completeness we present the principle of optimatily in the form relevant to the path planning problem in this paper. Following Cooke and Halsey [5] and Kaufman and Smith [11], we consider the expanded static multiple objective network in which the state incorporates the current location in the network (node) at the current time.

THEOREM 1. (Principle of Optimality for Static Multiple Objective Networks). *A nondominated path p, leaving node i at time $t \in S_T$ and reaching node N at or before time $t_0 + T$, has the property that for each node*

j lying on this path, a subpath p_1, that leaves node j at time $t_j \in S_T$, $t_j > t$, and arrives at node N at or before time $t_0 + T$, is nondominated.

Proof. By contradiction. Assume to the contrary that p_1 is dominated (not nondominated), i.e., there exists path p^* leaving node j at time t_j and arriving at node N at or before time $t_0 + T$, such that

$$[c(p^*)] \leqslant [c(p_1)]$$

and

$$[c(p^*)]_r < [c(p_1)]_r \qquad \text{for some} \quad r \in \{1, ..., m\}. \tag{1}$$

Let p_2 be a subpath that leaves node i at time t and arrives at node j at time t_j. Thus we have two paths from node i to node N such that their total cost is, respectively,

$$[c(p_1)] + [c(p_2)]$$

and

$$[c(p^*)] + [c(p_2)].$$

Applying (1) we get

$$[c(p^*)] + [c(p_2)] \leqslant [c(p_1)] + [c(p_2)],$$

which implies that path $p = (p_2, p_1)$ consisting of subpath p_2 and p_1 is dominated.

By the principle of optimality [1] and Theorem 1, we establish that for $t \in S_T$,

$$\{[F_i(t)]\} = VMIN\{[c_{ij}(t)] + \{[F_j(t + [c_{ij}(t)]_1)]\}\}, \qquad i = 1, 2, ..., N-1, \tag{2}$$

$$\{[F_N(t)]\} = \{0\},$$

where operation *VMIN* computes vector costs of nondominated paths in the set being the algebraic sum of the cost vector $[c_{ij}(t)]$ and the set of vector costs of all nondominated paths that leave node j at time $t + [c_{ij}(t)]_1$. Computing all nondominated paths in $\{\text{Eff}(E_i(t))\}$ requires applying an iteration scheme on the system of equations above.

We present now Algorithm One which includes the iterative procedure and finds $\{\text{Eff}(E_i(t))\}$, $i = 1, 2, ..., N-1$, in a finite number of steps.

ALGORITHM ONE.

Step 1. Establish a limited grid of discrete values of time $S_T = \{t_0, t_0+1, t_0+2, ..., t_0+T\}$. (Choice of T is discussed above). For $t \in S_T$ and $i, j = 1, 2, ..., N$, $i \neq j$, compute $[c_{ij}(t)]$.

Step 2. Modify the vectors $[c_{ij}(t)]$, $t \in S_T$ as follows:

$$[c_{ij}(t)]' = \begin{cases} [c_{ij}(t)] & \text{if } t + [c_{ij}(t)]_1 \leq t_0 + T \\ [\infty] & \text{if } t + [c_{ij}(t)]_1 > t_0 + T \end{cases} \qquad i, j = 1, 2, ..., N, i \neq j. \tag{3}$$

Step 3. Construct an "initial guess" array $[\{[F_i(t)^{(0)}]\}]$, $i = 1, 2, ..., N$, $t \in S_T$, where $\{[F_N(t)^{(0)}]\} = \{0\}$, and $\{[F_i(t)^{(0)}]\} = [c_{iN}(t)]'$ for $i = 1, 2, ..., N-1$.

Step 4. Calculate the arrays $[\{[F_i(t)^{(k)}]\}]$, $i = 1, 2, ..., N$, $t \in S_T$, for $k = 1, 2, 3, ...$ as follows:

$$\{[F_i(t)^{(k)}]\} = VMIN\{[c_{ij}(t)]' + \{[F_j(t + [c_{ij}(t)]'_1)^{(k-1)}]\}\}, \qquad i = 1, 2, ..., N-1, \tag{4}$$

$$\{[F_N(t)^{(k)}]\} = \{0\}.$$

The *VMIN* operation in the equation above will lead to $\{\infty\}$ if $[c_{ij}(t)]' = [\infty]$ for all j or if $t + [c_{ij}(t)]'_1 \notin S_T$. Otherwise, if both $[c_{ij}(t)]'$ and $\{[F_j(t + [c_{ij}(t)]'_1)^{(k-1)}]\}$ are finite for some j, compute $[c_{ij}(t)]'$, extract $\{[F_j(t + [c_{ij}(t)]'_1)^{(k-1)}]\}$ from the array $[\{[F_i(t)^{(k)}]\}]$ and perform the *VMIN* operation (over j) of their algebraic sum.

Step 5. The sequence of sets $\{[F_i(t_0)^{(k)}]\}$, $k = 1, 2, ...$ converges to $\{[F_i(t_0)]\}$. The set $\{\text{Eff}(E_i(t_0))\}$ is obtained by keeping track of the indices of paths' links that contribute to $\{[F_i(t_0)^{(k)}]\}$.

THEOREM 2. *The method of Algorithm One is* well-defined.

Proof. We start (step $k = 0$) with all paths and corresponding costs from node i to node N set equal to the single link costs connecting each node to N. That is, we start with nondominated paths of one link each. Each set $\{[F_i(t)^{(0)}]\} \neq \varnothing$, and contains exactly one cost vector. As k increases, the set $\{[F_i(t)^{(k)}]\}$ accumulates the nondominated cost vectors, while always dropping dominated cost vectors. Hence, $\{[F_i(t)^{(k)}]\}$ remains nonempty for any k. Therefore the *VMIN* operation may be applied for any k.

THEOREM 3. *The iterative step of Eq.* (4) *computes the set of all vector costs corresponding to all nondominated paths with at most k links connecting node i to node N with departure time t.*

Proof. The proof is by induction on k. Assume that $k=1$ and define for $t \in S_T$:

$$\{[F_i(t)^{(1)}]\} = \begin{cases} VMIN \text{ over the vector costs of paths in } \{E_i(t)^{(2)}\}, \\ \qquad \text{if} \quad \{E_i(t)^{(2)}\} \neq \varnothing, \\ \{\infty\} \qquad \text{if} \quad \{E_i(t)^{(2)}\} = \varnothing, \end{cases}$$
$$i = 1, 2, \ldots, N-1,$$

$$\{[F_N(t)^{(1)}]\} = \{0\}.$$

If $\{E_i(t)^{(2)}\} = \varnothing$, then the one-link path from node i to node N, leaving at time t reaches node N after time $t_0 + T$, so that $[c_{ij}(t)]' = [\infty]$. There is also no two-link path from node i through node j to node N that reaches node N by time $t_0 + T$, so that no one-link path leaving node j at time $t + [c_{ij}(t)]'_1$ reaches node N by time $t_0 + T$. Therefore either $[c_{ij}(t)]' = [\infty]$ or $\{[F_j(t + [c_{ij}(t)]'_1)^{(0)}]\} = \{\infty\}$ and $\{[F_i(t)^{(1)}]\} = \{\infty\}$.

If $\{E_i(t)^{(2)}\} \neq \varnothing$, then there exists at least one nondominated path of one or two links leaving node i at time t and reaching node N by time $t_0 + T$. A one-link nondominated path may have been obtained from initialization and still be nondominated. A two-link nondominated path leads from node i to node j, and then follows the one-link path from node j to node N with the arrival time at node j equal to $t + [c_{ij}(t)]'_1 = t + [c_{ij}(t)]_1$.

Now assume that the iterative step is valid for $k > 1$ and we will show its validity for $k+1$. Again define for $t \in S_T$:

$$\{[F_i(t)^{(k+1)}]\} = \begin{cases} VMIN \text{ over the vector costs of paths in } \{E_i(t)^{(k+2)}\}, \\ \qquad \text{if} \quad \{E_i(t)^{(k+2)}\} \neq \varnothing, \\ \{\infty\} \qquad \text{if} \quad \{E_i(t)^{(k+2)}\} = \varnothing, \end{cases}$$
$$i = 1, 2, \ldots, N-1,$$

$$\{[F_N(t)^{(k+1)}]\} = \{0\}.$$

If $\{E_i(t)^{(k+2)}\} = \varnothing$, then there is no path of at most $k+2$ links that leaves node i at time t and reaches node N by time $t_0 + T$. Therefore $\{[F_i(t)^{(k+1)}]\} = \{\infty\}$.

If $\{E_i(t)^{(k+2)}\} \neq \varnothing$, then there exists at least one nondominated path of at most $k+2$ links leaving node i at time t and reaching node N by time $t_0 + T$. A nondominated path of at most $k+1$ links leaving node i at time

t and reaching node N by time $t_0 + T$ may have been obtained in the kth or an earlier iteration of the algorithm. In the $k+1$ iteration one may obtain a nondominated path of at most $k+2$ links leading from node i to node j, and then following a path of at most $k+1$ links from node j to node N, that has been found in the kth or an earlier iteration on the algorithm.

THEOREM 4. *After a finite number of steps Algorithm One generates* all *nondominated paths that leave node* i, $i = 1, 2, ..., N-1$, *at time* t_0 *and reach node* N.

Proof. By contradiction. Assume that the method generated all nondominated paths, that leave node i at time t_0 and reach node N, except one. Then either the set of nondominated paths is incomplete or it includes at least one path that is dominated by the missing path. The former implies that the algorithm missed one nondominated path which contradicts the fact that each node of the network was visited and all links leading to it were examined. The latter indicates that the procedure did not identify the true status of a path, namely dominated. Hence arises a contradiction with performing the *VMIN* operation at node i, $i = 1, 2, ..., N-1$, of the network and discovering all nondominated paths leaving this node and reaching node N.

2.2. *Forward Dynamic Programming Case*

In the dynamic programming literature two problem formulations are commonly considered. For the multiple objective network there are: (1) find all nondominated paths from every node in the network to the destination node, or (2) from the origin node find all nondominated paths to every node in the network. While the former formulation gave rise to the development of Algorithm One, the latter will be considered in this section. The theoretical background of the multiple criteria dynamic network analysis will now be adapted for the second formulation. We relate the second formulation to the first as follows: the destination node is still the main focus. The forward solution is to be obtained with each other node in the network as the origin, and nondominated paths which reach the destination node are computed. The extra computations required may be compensated by other structural simplifications of the forward approach.

In this forward approach, feasible paths are those which start at the origin node. Assume that time t is a continuous variable, that is, $t \geqslant 0$, and hence allow $[c_{ij}(t)]_1$ to take any positive value. We normalize so that the departure time from the origin node is $t = 0$. Finally an assumption is introduced which allows the formulation of the principle of optimality for dynamic multiple objective networks.

Assumption 1. For any link (i, j) in the network and all $t_1, t_2 \geq 0$, if $t_1 \leq t_2$, then

(a) $t_1 + [c_{ij}(t_1)]_1 \leq t_2 + [c_{ij}(t_2)]_1$, and

(b) $[c_{ij}(t_1)]_r \leq [c_{ij}(t_2)]_r$ for all $r \in \{2, ..., m\}$.

We also introduce the following sets defined for $j = 2, 3, ..., N$, and vectors defined for $t > 0$ and $j = 2, 3, ..., N$:

$\{D_j\}$—the set of all paths in the network which leave the origin node at time $t = 0$ and lead to node j;

$\{\text{Eff}(D_j)\}$—the set of all nondominated paths which leave the origin node at time $t = 0$ and lead to node j;

$\{D_j^{(k)}\}$—the set of all paths of at most k links leaving the origin node at time $t = 0$ and leading to node j, $k = 2, 3, ...$;

$[G_j^u(t^u)^{(k)}]$—the vector cost of the nondominated path u in $\{D_j^{(k+1)}\}$, where t^u is the arrival time of this path at node j;

$\{[G_j^{(k)}]\} = \{[G_j(t^u)^{(k)}],\ u = 1, ..., N_j\}$—the set of vectors costs of all nondominated paths in $\{D_j^{(k+1)}\}$, where N_j is the number of the nondominated paths;

$[G_j^u(t^u)]$—the vector cost of the nondominated path u leaving the origin node at time $t = 0$ and arriving at node j at time t^u;

$\{[G_j]\} = \{[G_j(t^u)], u = 1, ..., N_j\}$—the set of vector costs of all nondominated paths in $\{\text{Eff}(D_j)\}$, where N_j is the number of the nondominated paths.

Obviously, $\{\text{Eff}(D_j)\} \subseteq \{D_j\}$.

THEOREM 5 (Principle of Optimality for Dynamic Multiple Objective Networks). *Under Assumption* 1(a) *and* (b), *a nondominated path p, that leaves the origin node at time* $t = 0$ *and arrives at node j at time* t_j, *has the property that for each node i lying on this path, a subpath* p_1, *that leaves the origin node at time* $t = 0$ *and arrives at node i at time* t_i, $t_i \leq t_j$, *is nondominated.*

Proof. By contradiction. Assume to the contrary that p_1 is dominated (not nondominated), i.e., there exists path p^* that leaves the origin node at time t and arrives at node i at time $t_i^* \leq t_i$ such that

$$[c(p^*)] \leq [c(p_1)], \tag{5}$$

and

$$[c(p^*)]_r < [c(p_1)]_r \qquad \text{for some} \quad r \in \{1, ..., m\}.$$

Both paths, p_1 and p^*, leave the origin node at the same time $t=0$ and lead to node i. Hence there are two paths from node i to node j such that one of them leaves node i at time t_i and arrives at node j at time t_j, and the other leaves node i at time t_i^* and arrives at node j at time t_j^*. Let us call those paths p_2 and p_2^*, respectively. Thus we have two paths from the origin node to node j such that their total cost is respectively:

$$[c(p_1)]+[c(p_2)]=[c(p_1)]+\sum_{\substack{(k,l)\in p_2\\ k=i}}^{l=j}[c_{kl}(t_k)] \tag{6}$$

and

$$[c(p^*)]+[c(p_2^*)]=[c(p^*)]+\sum_{\substack{(k,l)\in p_2^*\\ k=i}}^{l=j}[c_{kl}(t_k^*)], \tag{7}$$

where t_k and t_k^* are the arrival times at node k $(k=i, ..., l=j)$ on paths p_2 and p_2^*, respectively.

We also have that $t_i^* \leqslant t_i$, then by Assumption 1(a)

$$t_i^*+[c_{is}(t_i^*)]_1 \leqslant t_i+[c_{is}(t_i)]_1$$

for every link (i, s) in the network. Applying Assumption 1(a) on every link (k, l) in path p_2 and p_2^*, and substituting arrival times at each node, we reach node j and get

$$t_i^*+\sum_{\substack{(k,l)\in p_2^*\\ k=i}}^{l=j}[c_{kl}(t_k^*)]_1 \leqslant t_i+\sum_{\substack{(k,l)\in p_2\\ k=i}}^{l=j}[c_{kl}(t_k)]_1,$$

which by definition means

$$t_j^* \leqslant t_j. \tag{8}$$

Applying Assumption 1(b) and summing over all links on p_2 and p_2^* we have

$$\sum_{\substack{(k,l)\in p_2^*\\ k=i}}^{l=j}[c_{kl}(t_k^*)]_r \leqslant \sum_{\substack{(k,l)\in p_2\\ k=i}}^{l=j}[c_{kl}(t_k)]_r, \qquad r\in\{2, ..., m\}. \tag{9}$$

Note that inequalities (5), (8), and (9) lead to

$$[c(p^*)]+[c(p_2^*)] \leqslant [c(p_1)]+[c(p_2)],$$

which implies that path $p=(p_1, p_2)$ consisting of subpaths p_1 and p_2 is dominated.

By the principle of optimality [1] and Theorem 5, we establish that for $t^\ell > 0$ and $t^n > 0$:

$$\{[G_j^\ell(t^\ell)], \ell = 1, ..., N_j\} = VMIN\{[G_i^n(t^n)] + [c_{ij}(t^n)], \quad n = 1, ..., N_i\}, \quad j = 2, 3, ..., N, \tag{10}$$

$$\{[G_j^\ell(t^\ell)], \ell = 1\} = \{0\},$$

where operation *VMIN* computes vector costs of nondominated paths in the set for which each element is a vector sum of the vector cost of the nondominated path n leaving the origin node at time 0 and arriving at node i at time t^n, and the cost vector of link (i, j) with the arrival time t^n at node i. Computing all nondominated paths in $\{\text{Eff}(D_j)\}$ requires again applying an iteration scheme on the system of equations above.

We now present Algorithm Two which includes the iterative procedure and finds $\{\text{Eff}(D_j)\}$, $j = 2, 3, ..., N$, in a finite number of steps. (Assume without loss of generality that node 1 is the origin node.)

ALGORITHM TWO.

Step 1. Construct an "initial guess" vector $[\{[G_j^{(0)}]\}], j = 1, 2, ..., N$, where

$$\{[G_1^{(0)}]\} = \{0\},$$
$$\{[G_j^{(0)}]\} = [c_{1j}(0)], \quad j = 2, 3, ..., N.$$

Step 2. Calculate the vectors $[\{[G_j^{(k)}]\}]$, $j = 1, 2, ..., N$, for $k = 1, 2, 3, ...$, as follows:

$$\{[G_j^\ell(t^\ell)^{(k)}], \ell = 1, ..., N_j\} = VMIN\{[G_i^n(t^n)^{(k-1)}] + [c_{ij}(t^n)], n = 1, ..., N_i\}, \quad j = 2, 3, ..., N,$$

$$\{[G_1^\ell(t^\ell)^{(k)}], \ell = 1\} = \{0\}. \tag{11}$$

The *VMIN* operation in the equation above will lead to $\{\infty\}$ if $[c_{ij}(t^n)] = [\infty]$ for all nondominated paths leading to node i $(n = 1, ..., N_i)$. Otherwise, if both $[c_{ij}(t^n)]$ and $[G_i^n(t^n)^{(k-1)}]$ are finite for some nondominated path n, then compute their vectors sum for each such path, and perform the *VMIN* operation (over i) on the set just obtained.

Step 3. The sequence of sets $\{[G_j^{(k)}]\}$, $k = 1, 2, ...$, converges to $\{[G_j]\}$. The set $\{\text{Eff}(D_j)\}$ is obtained by keeping track of paths' links that contribute to $\{[G_j^{(k)}]\}$.

The proofs of the following theorems are similar to the proofs of Theorems 2, 3, and 4, and are thus omitted.

THEOREM 6. *The method of Algorithm Two is* well defined.

THEOREM 7. *The iterative step of Eq.* (11) *computes the set of all vectors costs corresponding to all nondominated paths of at most k links connecting node* 1 *and node j with departure time* $t = 0$.

THEOREM 8. *After a finite number of steps Algorithm Two generates* all *nondominated paths that leave node* 1 *at time* $t = 0$ *and reach all other nodes j, for* $j = 2, 3, ..., N$.

COROLLARY 1. *If* $[c_{ij}(t)]_r$, $i, j = 1, 2, ..., N$, *and* $i \neq j$, $r = 1, 2, ..., m$, *is a continuous monotone increasing function on* $[0, \infty)$, *then Algorithm Two finds all nondominated paths from the origin node to any other node.*

Proof. A continuous monotone increasing function satisfies Assumption 1(a) and (b) and thus the Principle of Optimality for Dynamic Multiple Objective Networks holds, and by Theorem 8 all nondominated paths from the origin node to any other node are generated.

COROLLARY 2. *If* $[c_{ij}(t)]_r$, $i, j = 1, 2, ..., N$, *and* $i \neq j$, $r = 1, 2, ..., m$, *is a monotone increasing step function on* $[0, \infty)$, *then Algorithm Two finds all nondominated paths from the origin node to any other node.*

Proof. Proof follows the proof of Corollary 1 since a monotone increasing step function satisfies Assumption 1(a) and (b).

COROLLARY 3. *If* $[c_{ij}(t)]_r$, $i, j = 1, 2, ..., N$, *and* $i \neq j$, $r = 1, 2, ..., m$, *is a continuous monotone increasing function on* $[0, \infty)$, *then all nondominated paths from node* i, $i = 1, 2, ..., N-1$, *starting at time* $t = 0$ *and leading to the destination node N may be computed with at most* $N-1$ *applications of Algorithm Two.*

COROLLARY 4. *If* $[c_{ij}(t)]_r$, $i, j = 1, 2, ..., N$, *and* $i \neq j$, $r = 1, 2, ..., m$, *is a monotone increasing step function on* $[0, \infty)$, *then all nondominated paths from node* i, $i = 1, 2, ..., N-1$, *starting at time* $t = 0$ *and leading to the destination node N may be computed with at most* $N-1$ *applications of Algorithm Two.*

As it was mentioned above, computing all nondominated paths from any node to the destination node is the main interest. Corollaries 3 and 4, that result immediately from Corollaries 1 and 2 and thus are presented without proofs, show that the forward approach can be applied to solving that problem. Although such a method becomes more complex computationally

(at most $(N-1)$ Algorithm Two solutions), it is competitive with Algorithm One, which requires storing and computing a large amount of data for the expanded static multiple objective network.

3. Examples

The algorithms presented in the previous section are now applied to solve two dynamic routing problems. The notation in each of the subsections below agrees with symbols previously used in Sections 2.1 and 2.2, respectively.

3.1. *Backward Dynamic Programming Case*

We will use Algorithm One to solve a dynamic routing problem with two criteria ($m=2$) for the network dipicted in Fig. 1. The example is related to a somewhat simpler example in Kaufman and Smith [11]. Remarks there indicate that a naive approach will fail on their example. Similarly, one needs the expanded static network to solve the network.

A grid of discrete values of time $S_{19}=\{1, 2, ..., 20\}$ for $t_0=1$ is established, and vectors $[c_{ij}(t)]$ for $t\in S_{19}$ are modified according to Step 2 of the algorithm, which results in the expanded static network. An initial guess array $[\{[F_i(t)^{(0)}]\}]$, $i=1, ..., 4$, $t\in S_{19}$, is constructed, and the arrays $[\{[F_i(t)^{(1)}]\}]$ and $[\{[F_i(t)^{(2)}]\}]$ are calculated as Step 4 of the algorithm indicates. Figure 2 shows the initial array and the two subsequent arrays calculated in Step 4 of the algorithm (each array has 20 rows and 4 columns that are shown in Fig. 2 in a truncated form). $[\{[F_i(t_0)^{(2)}]\}]$ is given by the first row of array $[\{[F_i(t)^{(2)}]\}]$ and contains vector costs of all nondominated paths that leave node i, $(i=1, ..., 4)$, at time $t_0=1$ and reach node 4.

The sets $\{\text{Eff}(E_i(t_0))\}$ of all nondominated paths that leave node i, $i=1, 2, 3$, at time $t_0=1$ are given as

$$\{(1, 2), (2, 3), (3, 4)\},$$

$$\{(2, 3), (3, 4)\},$$

$$\{(3, 4)\},$$

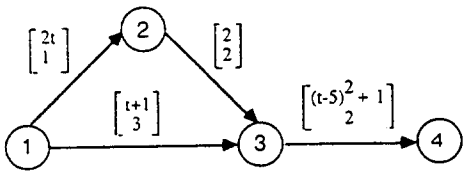


FIG. 1. Two-criteria dynamic network.

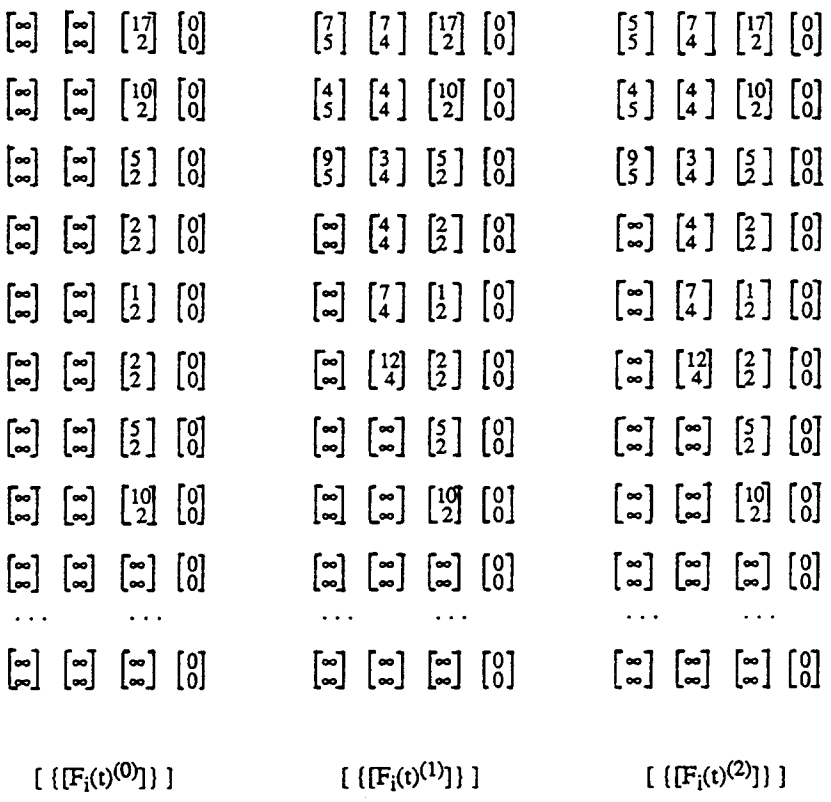


FIG. 2. Algorithm One, Step 4 calculations.

and are obtained by keeping track of the indices of paths' links that contribute to $[\{[F_i(t_0)^{(2)}]\}]$.

Observe that because the cost functions are not all monotone increasing functions of time, it is possible to have total cost behave in a non-monotonic way. In such a network, one may also experience "passing" by which one traveling unit overtakes another on a link.

3.2. *Forward Dynamic Programming Case*

We solve a two-criteria dynamic routing problem presented by the network in Fig. 3 and apply Algorithm Two. Cost functions here satisfy Assumption 1. Therefore, a "no-passing" convention is in effect. For comparison, the cost functions are chosen as a combination of constant functions and monotone increasing step functions.

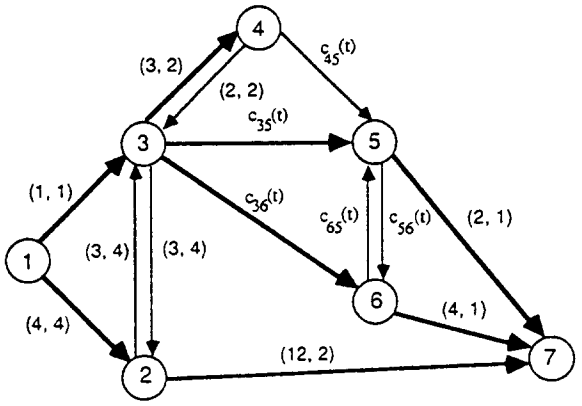


FIG. 3. Two-criteria dynamic network with step cost functions and its nondominated paths.

The step cost functions are given as

$$c_{35}(t)=\begin{cases}\begin{bmatrix}6\\8\end{bmatrix} & t<3\\ \begin{bmatrix}10\\12\end{bmatrix} & t\geqslant 3\end{cases}, \qquad c_{45}(t)=\begin{cases}\begin{bmatrix}4\\4\end{bmatrix} & t<3\\ \begin{bmatrix}10\\10\end{bmatrix} & t\geqslant 3\end{cases}, \qquad c_{65}(t)=\begin{cases}\begin{bmatrix}8\\10\end{bmatrix} & t<3\\ \begin{bmatrix}10\\12\end{bmatrix} & t\geqslant 3\end{cases},$$

$$c_{36}(t)=\begin{cases}\begin{bmatrix}5\\7\end{bmatrix} & t<5\\ \begin{bmatrix}10\\12\end{bmatrix} & t\geqslant 5\end{cases}, \qquad c_{56}(t)=\begin{cases}\begin{bmatrix}5\\10\end{bmatrix} & t<5\\ \begin{bmatrix}12\\12\end{bmatrix} & t\geqslant 5\end{cases}.$$

All the other links of the network have constant vector costs, as Fig. 3 shows. We start with an initial guess vector $[\{[G_j^{(0)}]\}]$, $j=1, 2, ..., 7$, and calculate vectors $[\{[G_j^{(k)}]\}]$, $j=1, 2, ..., 7$, according to Step 2 of the algorithm. The vectors $[\{[G_j^{(k)}]\}]$, $j=1, 2, ..., 7$, for $k=0, 1, ..., 4$ are shown in Fig. 4.

The sequence of sets $\{[G_j^{(k)}]\}$ converges to $\{[G_j]\}$ in the second iteration of the algorithm. The sets $\{\text{Eff}(D_j)\}$ for $j=2, 3, ..., 7$, are obtained by keeping track of paths' links that contribute to $\{[G_j^{(k)}]\}$ and include the following paths:

$\{(1, 2)\}$,
$\{(1, 3)\}$,
$\{(1, 3), (3, 4)\}$,
$\{(1, 3), (3, 5)\}$,
$\{(1, 3), (3, 6)\}$,
$\{(1, 2), (2, 7)\}$, $\{(1, 3), (3, 5), (5, 7)\}$, $\{(1, 3), (3, 6), (6, 7)\}$.

	k=0	k=1	k=2	k=3
$\{[G_1^{(k)}]\}$	$\begin{bmatrix}0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\end{bmatrix}$	$\begin{bmatrix}0\\0\end{bmatrix}$
$\{[G_2^{(k)}]\}$	$\begin{bmatrix}4\\4\end{bmatrix}$	$\begin{bmatrix}4\\4\end{bmatrix}$	$\begin{bmatrix}4\\4\end{bmatrix}$	$\begin{bmatrix}4\\4\end{bmatrix}$
$\{[G_3^{(k)}]\}$	$\begin{bmatrix}1\\1\end{bmatrix}$	$\begin{bmatrix}1\\1\end{bmatrix}$	$\begin{bmatrix}1\\1\end{bmatrix}$	$\begin{bmatrix}1\\1\end{bmatrix}$
$\{[G_4^{(k)}]\}$	$\begin{bmatrix}\infty\\\infty\end{bmatrix}$	$\begin{bmatrix}4\\3\end{bmatrix}$	$\begin{bmatrix}4\\3\end{bmatrix}$	$\begin{bmatrix}4\\3\end{bmatrix}$
$\{[G_5^{(k)}]\}$	$\begin{bmatrix}\infty\\\infty\end{bmatrix}$	$\begin{bmatrix}7\\9\end{bmatrix}$	$\begin{bmatrix}7\\9\end{bmatrix}$	$\begin{bmatrix}7\\9\end{bmatrix}$
$\{[G_6^{(k)}]\}$	$\begin{bmatrix}\infty\\\infty\end{bmatrix}$	$\begin{bmatrix}6\\8\end{bmatrix}$	$\begin{bmatrix}6\\8\end{bmatrix}$	$\begin{bmatrix}6\\8\end{bmatrix}$
$\{[G_7^{(k)}]\}$	$\begin{bmatrix}\infty\\\infty\end{bmatrix}$	$\begin{bmatrix}16\\6\end{bmatrix}$	$\begin{bmatrix}16\\6\end{bmatrix}, \begin{bmatrix}9\\10\end{bmatrix}, \begin{bmatrix}10\\9\end{bmatrix}$	$\begin{bmatrix}16\\6\end{bmatrix}, \begin{bmatrix}9\\10\end{bmatrix}, \begin{bmatrix}10\\9\end{bmatrix}$

FIG. 4. Algorithm Two, Step 2 calcuations.

Note that this calculation produces the set of nondominated paths from node 1 to node 7 (destination node) as well as to all other nodes. A similar computation is required for nodes 2, 3, 4, 5, and 6 to compute all nondominated paths from these nodes to the destination node. Applying Algorithm Two for each of these nodes gives the following nondominated paths:

$\{(2, 7)\}$,

$\{(3, 2), (2, 7)\}$, $\{(3, 6), (6, 7)\}$, $\{(3, 5), (5, 7)\}$,

$\{(4, 5), (5, 7)\}$,

$\{(5, 7)\}$,

$\{(6, 7)\}$.

4. CONCLUSIONS

This paper presents for the first time a theoretical and algorithmic development for the problem of path planning in networks including multiple time dependent costs on the links. Throughout, the goal is to compute all nondominated paths in an efficient computational procedure. Applications abound in fire safety science, general transportation, and telecommunications.

Our study has produced two distinct algorithms, each one tailored to a particular class of cost functions. For general costs functions, we extend the work of Cooke and Halsey [5] to handle multiple objective functions. We observe that their paper solved only the minimum travel time path planning problem, so this paper seems to be the first to handle cost functions other than travel time. This algorithm should be used only when required (by cost functions which are not monotone increasing) because it has a greater overhead and computational cost than the other algorithms.

For monotone increasing cost functions satisfying one additional assumption, a forward dynamic programming algorithm which generalizes the paper of Kaufman and Smith [11] is presented. Such an algorithm seems more computationally effective than the one for general cost functions and it seems to be independent of the time horizon with repect to its computational complexity. It has the disadvantage that it must be applied to each origin node independently in order to get a complete set of nondominated paths.

ACKNOWLEDGMENTS

This research was supported in part by Grant 60NANB0D1023 from the Building and Fire Research Laboratory, National Institute of Standards and Technology, Gaithersburg, Maryland.

REFERENCES

1. R. BELLMAN, On a routing problem, *Quart. Appl. Math.* **16** (1958), 87–90.
2. D. BERTSEKAS AND R. GALLAGER, "Data Networks," Prentice–Hall, Englewood Cliffs, NJ, 1987.
3. T. A. BROWN AND R. E. STRAUCH, Dynamic programming in multiplicative lattices, *J. Math. Anal. Appl.* **12** (1965), 364–370.
4. R. L. CARRAWAY AND T. L. MORIN, Theory and applications of generalized dynamic programming: An overview, *Comput. Math. Appl.* **16**, No. 10/11 (1988), 779–788.
5. K. L. COOKE AND E. HALSEY, The shortest route through a network with time-dependent internodal transit times, *J. Math. Anal. Appl.* **14** (1966), 493–498.
6. H. W. CORLEY AND I. D. MOON, Shortest paths in networks with vector weights, *J. Optim. Theory Appl.* **46** (1985), 79–86.
7. H. G. DAELLENBACH AND C. A. DE KLUYVER, Note on multiple objective dynamic programming, *J. Oper. Res. Soc.* **31** (1980), 591–594.
8. J. R. EVANS, C. SAYDAM, AND M. MCKNEW, A note on solving the concave cost dynamic lot-sizing problem in almost linear time, *J. Oper. Management* **8**, No. 2 (1989), 159–167.
9. R. HARTLEY, Vector optimal routing by dynamic programming, *in* "Mathematics of Multiobjective Optimization" (P. Serafini, Ed.) pp. 215–224, Springer-Verlag, Vienna, 1985.
10. T. IBARAKI, Enumerative approaches to combinatorial optimization, Part II, *Ann. Oper. Res.* **11** (1987), 343–440.

11. D. E. KAUFMAN AND R. L. SMITH, "Minimum Travel Time Paths in Dynamic Networks with Application to Intelligent Vehicle–Highway Systems," University of Michigan, Transportation Research Institute, IVHS Technical Report-90-11, 1990.
12. M. M. KOSTREVA, M. M. WIECEK, AND T. GETACHEW, Optimization models in fire egress analysis for residential buildings, *in* "Proceedings of the Third International Symposium on Fire Safety Science, Edinburgh, United Kingdom, July 8–12, 1991," 805–814, (updated).
13. D. LI AND Y. Y. HAIMES, Multiobjective dynamic programming: The state of the art, *Control Theory Adv. Tech.* **5**, No. 4 (1989), 471–483.
14. A. ORDA AND R. ROM, Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length, *J. A. C. M.* **37**, No. 3 (1990), 607–625
15. A. G. PEREVOZCHIKOV, Dynamic programming in multistep vector optimization problems, *Eng. Cybernet.* **22**, No. 3 (1984) 21–24.
16. H.-J. SEBASTIAN, Dynamic programming for problems with time-dependent parameters, *Differential Equations* **14**, No. 2 (1978), 242–249.

Printed by Catherine Press, Ltd., Tempelhof 41, B-8000 Brugge, Belgium

Appendix 7

"Pareto Optimization in Dynamic Networks with Bounded Cost Functions," submitted for publication (with T. Getachew).

Pareto Optimization in Dynamic Networks with Bounded Cost Functions

by

Teodros Getachew
and
Michael M. Kostreva

Department of Mathematical Sciences
Clemson University
Clemson, South Carolina 29634-1907

1. INTRODUCTION

Ever since its formulation by Bellman [2], Dynamic Programming has shown itself to be a useful tool in the solution of multi-stage decision problems. Over the last few decades, there have been numerous attempts to generalize it, notably:

1. with regard to the space of objectives,

2. with regard to the dimensionality of the cost vectors, and

3. with regard to the time-dependency of the cost vectors.

As this paper proposes a new addition to this body of knowledge, it will commence with a review of the historical background of this progression of generalizations.

In an early paper, Brown and Strauch [4] generalize Dynamic Programming by allowing the range of objective functions to be a regular multiplicative lattice. In similar work , Henig [15] investigates the Principle of Optimality when the objective values are in a partially ordered set. Verdu and Poor [20] propose an abstract Dynamic Programming model that includes, but is not restricted to, optimization problems.

Daellenbach and De Kluyver [8] introduce a computational method for the determination of Pareto Optimal paths with the restriction that the distance from a given node to the destination node is unique. The method they propose, emulated in its basic form in many of the papers to follow on the subject, is a straight-forward extension of the Principle of Optimality to the multiple objective context. Corley and Moon [7] consider the same problem as Daellenbach and Kluyver, and use the same algorithm, with the vector-minimization taking place

over paths of length k or fewer links, thus allowing for the possibility that a given node can be the initial node of paths of varying lengths.

An early paper to consider time dependency in Dynamic Programming is Cooke and Halsey [6]. The case they consider is the routing problem of finding minimum travel times to a destination, where the transition times between states are themselves time dependent. This method has been extended to the multi-criteria case by Kostreva and Wiecek [17]. In a general survey of shortest-path algorithms, Dreyfus [11] briefly discusses the problem of finding shortest paths in networks with time-dependent length of arcs. He proposes a modification of Dijkstra's [10] algorithm as a method of solution.

Halpern [14] proposes an algorithm to determine the shortest route in a network with edge transit times that are time-varying, and with limited delay at nodes. In work related to that of Halpern, Orda and Rom [18] consider the same problem under numerous waiting models. They discuss the computational complexity of their proposed algorithms. Brumbaugh and Shier [5] note that the problem of finding efficient paths in bicriteion networks is, in general, exponential in the network size, in contrast to the single criterion case for which polynomial algorithms exist. They carry out an empirical investigation of some label correcting algorithms devised to solve this problem. Philpott [19] attacks a similar problem using a continuous-time linear programming formulation.

Finally, Kostreva and Wiecek [17] generalize earlier work by Kaufman and Smith [16] on finding minimum travel time paths in networks with time-varying transit times. The generalization solves the problem of finding all nondominated paths from the origin node to all other nodes in the network when the cost functions on the links satisfy monotonicity and passing is not allowed.

There exist counterexamples to Halpern's and Dreyfus' algorithms for the solution of the shortest path problem with time dependent costs [12]. These

algorithms fail because they are "memoryless". That is, they have no provision for recovering subpaths that were discarded because they were nonoptimal locally, but which nevertheless were part of optimal paths, (optimal paths which the algorithms , based on link augmentation, no longer may access, their subpaths having been discarded earlier). Cooke and Halsey's algorithm avoids this difficulty precisely because it does not discard information; information with regard to optimal paths emanating from a node, for the given set of start times, is stored, and accessed as necessary. The shortcomings of the Cooke and Halsey algorithm lie in its prohibitive demands on memory (time expanded network), especially in its multi-objective version, and its restriction with regard to the integrality of the transit times.

The algorithm that is presented in this paper has the following features:

1. It is recursive,
2. It takes place both in the set of links, in a backward Dynamic Programming phase, and the set of paths, in a forward evaluation phase,
3. The algorithm is free from Cooke and Halsey type restrictions on the integrality of transit times, and the associated large network,
4. The algorithm requires only boundedness of the cost functions; this extends the results of Kostreva and Wiecek, whose algorithm required monotonicity of costs,
5. It reduces to backward multi-objective Dynamic Programming if costs are constant.

2. A DESCRIPTION OF THE ALGORITHM

Suppose one is given a set of nodes joined by links, with each link having associated with itself a vector of functions that specifies the cost of making the

transition across it, functions that are dependent on the arrival time at the initial node of the link. Find among those continuous, non self-crossing paths from every node to a designated destination node, all the efficient ones with respect to the cumulative transition cost vector.

The algorithm commences with a time-invariant network in which each link has a transition cost, possibly vector-valued, that coincides with the infimum of the link transition cost function. The set of efficient paths from all nodes to a destination node is then determined by backward Dynamic Programming. This ends phase one of the algorithm for the initial iteration.

Phase two begins with the determination of the true costs for all members of this efficient set. This is done by direct recursive evaluation, as follows. The time of traversing the initial link of a given path is equal to the value of the transition time of that link, evaluated at the initial time, which without loss of generality will be t = 0; assuming the times of arrival at earlier nodes have been calculated for the first k nodes in a path, the cumulative transition time to the $(k+1)^{st}$ node is calculated by adding the arrival time at the k^{th} node to the transition time of the link of which the k^{th} node is the initial and the $(k+1)^{st}$ the final one. This latter transition time is calculated by evaluating the transition cost of this link at t equal to the arrival time at the kth node. The time to traverse the entire path is thus calculated. The other components of the cumulative cost vector are calculated in the same manner, using the appropriate transition cost functions evaluated at the appropriate arrival times. Now, for each efficient path with start node r, a comparison is made between the (infimum) costs of the efficient paths calculated in phase one, and the (true) costs calculated at the beginning of phase two. Consider now the set of paths for which these two costs are unequal. For all those start nodes for which this set is empty the set of all efficient paths from the given node to the destination node has been determined.

It is the set of efficient paths with start node the given node, found at the end of phase one. Otherwise define, for each start node, the set of avoidance paths to consist of those paths for which the infimum cost differs from the actual cost, together with the set subpaths of these paths that end in the destination node.

Suppose now that the avoidance set has been defined for iterations up to and including r-1. Iteration r is executed as follows: Through backward Dynamic Programming, determine the efficient set over the (time invariant) infimum costed network consisting of paths not in the avoidance set defined in iteration r-1. This is done in the following manner. Suppose all efficient paths that are in the complement of the avoidance set and have length at most k-1 have been determined. A non-avoidance path of length k from a given node can arise in two and only two ways; as a path with an efficient path of length k-1, or a path with a avoidance path of length k-1. In the first case, the path of length k is constructed by a one-link augmentation of a (k-1)-long efficient path from the just completed Dynamic Programming stage, and in the second, by a one-link augmentation of a (k-1)-long avoidance path which has been determined in the previous iteration of the algorithm. Phase one ends with a vector minimization over the union of the set of efficient paths just determined with the set of avoidance paths determined in the previous iteration.

Phase two for this iteration begins with the determination of the true costs of the efficient set determined by the vector minimization at the end of phase one, precisely as in the initial iteration. Again, consider the set of efficient paths for which the infimum cost differs from the true cost. All those start nodes for which this set is empty have just had their entire efficient set determined at the end of phase one of the present iteration. For each node such that these two costs differ, however, form the new avoidance set: as the union of the set of

paths whose infimum costs differ from their true costs with the avoidance set from the previous iteration. Proceed to iteration r+1.

A rigorous development of the algorithm follows in section 3.

3. THEORETICAL DEVELOPMENT

Notation, Definitions and Terminology

The setting for the development of the algorithm is a network.

Definition 1: Let **N**, a set of n distinct points, called nodes, and L, a set of ordered pairs of distinct elements from **N**, called links, be given. A directed network with node set **N** and link set L is,

$$G = (N, L).$$

Definition 2: An element (i,j) of L is referred to as a directed link from node i to node j.

Definition 3: A set π of elements of directed links is said to be a path from i to j, $i \neq j$, iff

$$\pi = \{ (i, j_1), (j_1, j_2), \ldots, (j_{k-2}, j_{k-1}), (j_{k-1}, j) \}$$

π is said to have cardinality k.

In this paper, we shall only be interested in paths that terminate in a pre-selected node. The appropriate notation is given below. Let a distinguished element d of N, called the destination node be given. Then $P_i(G)$ shall denote the set of paths π with initial node i and final node d.

The set of all such paths, emanating from nodes that are distinct from the destination node, will be defined by the equation:

$$P(G) = \bigcup_{i \in N \setminus \{d\}} P_i(G)$$

In the first phase of the algorithm, it is important to isolate those paths satisfying specific cardinality criteria. Accordingly, some notation is given next.

Notation 1: $P_i^{(k)}(G)$ shall denote the set of paths in $P_i(G)$ with each path of cardinality at most k.

Notation 2: $P_i^{k}(G)$ shall denote the set of paths in $P_i(G)$ with each element of cardinality exactly k.

Note then that the entirety of paths of length at most k is defined by the equation

$$P^{(k)}(G) = \bigcup_{i \in N \setminus \{d\}} P_i^{(k)}(G)$$

and the set of all paths of length exactly k by

$$P^{k}(G) = \bigcup_{i \in N \setminus \{d\}} P_i^{k}(G).$$

The concept of a subpath is fundamental , particularly in phase one of the algorithm, which, through backward Dynamic Programming, constructs paths from their subpaths, by link augmentation.

Definition 4: A subset π' of a path $\pi \in P(G)$ is said to be a subpath of π if and only if $\pi' \in P(G)$. That is, a subpath, in addition to being a subset of a path, must itself have the destination node as its terminal node.

We next introduce time dependency. The functionality on time is quite unrestricted; it is not required to satisfy, for instance, continuity or monotonicity conditions. The only restriction on the functions that define the costs on links is that they be bounded. Boundedness above is imposed for the global requirement of finite costing, while boundedness below is a requirement of the first phase of the algorithm. Moreover, the costs of subpaths do not have to be separable from the costs of the paths that contain them.

Definition 5: Let F be a set of functions with domain $R^+ \cup \{0\}$ and range R^+, bounded above and below on bounded intervals. Then the link-transition cost functions are given by the range of the function T where $T : L \rightarrow F$, given by

$$T((i,j)) = T_{ij}(t) \in F.$$

The cost on a link, as noted above, is a function, not only of time, but also of the paths to which it belongs. Given a particular path, and a particular link on it, the cost on the link is determined by the time of arrival, via the path, at the initial node of the link. This is formalized in terms of the arrival function.

Definition 6: Let $\pi = \{(i, j_1), (j_1, j_2), \ldots, (j_{k-2}, j_{k-1}), (j_{k-1}, d)\}$ be a path in $P_i^k(G)$. The arrival function A: $P(G) \times N \rightarrow R^+$ is defined recursively on initial link-nodes as follows:

$A(\pi,i) = 0$;

Suppose $A((\pi,j_{k-1}))$ has been defined for $r \leq s-1$; let (j_s, j_{s+1}) be an element of π. Then

$A(\pi, j_{s+1}) = A(\pi, j_s) + T(A(\pi, j_s))$.

The algorithm is developed in the context of multi-objective criteria where the cost on a link is a vector function of time, with each component a bounded function, as defined above. This is formalized in the:

Definition 7: C: $L \rightarrow F \times F \times \ldots \times F$ such that,

$C(i,j) = (c_{ij}^{(1)}(t), \ldots, c_{ij}^{(p)}(t))$, where each $c_{ij}^{(k)} \in F$, $k = 1, \ldots, p$..

The backward Dynamic Programming phase and the forward integration phase use different aspects of the link costs, in order to evaluate the total cost of a path. In the former, the infima of the link transition cost-functions are used, while in the latter their actual costs are determined. Accordingly, we next define the "path costing" functions Γ_o and Γ.

Definition 8: Let

$$c_o^{(k)}(n_i, n_j) = \inf_{t \in I(n_i, n_j)} c_{n_i n_j}^{(k)}(t) \quad \text{for } k = 1, 2, \ldots, p,$$

where,

$$l(n_i,n_j) = [\alpha,\beta],$$

with

$\alpha = \min A(\pi,n_i)$ and $\beta = \max A(\pi,n_j)$, for

$(n_i,n_j) \in \pi$.

Now, let $\pi \in P_i(G)$ be of cardinality k.

Then $\Gamma_o: P(G) \rightarrow R^+ \times R^+ \times \ldots \times R^+$ is defined by

$$\Gamma_o(\pi) = \vec{c}_0(i,j_1) + \vec{c}_0(j_{k-1},d) + \sum_{s=1}^{k-2}(\vec{c}_0(j_s,j_{s+1})),$$

where,

$$\vec{c}_0(j_q,j_s) = (c_o^{(1)}(j_q,j_s), \ldots, c^{(p)}(j_q,j_s)).$$

Next is the definition of the forward costing function:

Γ: $P(G) \rightarrow R^+ \times R^+ \times \ldots \times R^+$ is defined by,

$$\Gamma(\pi) = \vec{c}_{ij_1}(0) + \vec{c}_{j_{k-1}d}(A(\pi,j_{k-1})) + \sum_{s=1}^{k-2}\left(\vec{c}_{j_s j_{s+1}}(A(\pi,j_s))\right).$$

Finally, the notion of efficiency (Pareto optimality) is formally stated in terms of the notation given above.

Definition 9: Let $\pi \in P^{(k)}(G)$. Then, $\pi \in \text{eff}_i^{(k)}(P_o)$ if and only if the set $\{\pi': \pi' \in P^{(k)}(G)$ and $\Gamma_o(\pi') \leq \Gamma_o(\pi)$ and $\Gamma_o(\pi') \neq \Gamma_o(\pi)\}$ is empty. Similarly, $\pi \in \text{eff}_i^{(k)}(P)$ if and only if the set $\{\pi': \pi' \in P^{(k)}(G)$ and $\Gamma(\pi') \leq \Gamma(\pi)$ and $\Gamma(\pi') \neq \Gamma(\pi)\}$ is empty. Note that $\text{eff}(P_o) = \text{eff}_i^{(n-1)}(P_o)$ and $\text{eff}(P) = \text{eff}_i^{(n-1)}(P)$.

Initial Iteration I_o

Given a network, with each link having an associated transition cost function as defined above, the algorithm begins with an initial partition of the set of paths; namely, the partition consisting of the empty set and the entire set of

paths with all links costed as in Γ_o. Note that this costing yields a unique network, as each link transition cost function has a unique infimum. What follows then is an application of backward Dynamic Programming to this network.

$$F_{0i}^{(1)} = \text{VMIN}\{\Gamma_o(\pi) : \pi \in P_i^1(G)\};$$
$$F_{0i}^{(2)} = \text{VMIN}\{\Gamma_o(\pi): \Gamma_o(\pi) = \vec{c}_o(i,j) + f_{0j}^{(1)}, f_{0j}^{(1)} \in F_{0j}^{(1)}\};$$
$$\ldots$$
$$F_{0i}^{(r)} = \text{VMIN}\{\Gamma_o(\pi): \Gamma_o(\pi) = \vec{c}_o(i,j) + f_{0j}^{(r-1)}, f_{0j}^{(r-1)} \in F_{0j}^{(r-1)}\};$$
$$F_{0i}^{(n-1)} = \text{VMIN}\{\Gamma_o(\pi): \Gamma_o(\pi) = \vec{c}_o(i,j) + f_{0j}^{(n-2)}, f_{0j}^{(n-2)} \in F_{0j}^{(n-2)}\};$$

The backward Dynamic Programming stage terminates with the set $F_{0i}^{(n-1)}$. Let

$$F_{0i} \equiv F_{0i}^{(n-1)}, \text{ and}$$

$$P(F_{0i}) = \{\pi : \pi \in P(G) \text{ and } \Gamma_o(\pi) \in F_{0i}\}.$$

The true cost of each element whose cost is in the set $P(F_{0i})$ is now found by the evaluation, via the costing function Γ. If this cost differs from its cost under Γ_o, the path is appended to the set V_{0i}. Let

$$V_{0i} = \{\pi: \pi \in P(F_{0i}) \text{ and } \Gamma_o(\pi) \leq \Gamma(\pi) \text{ and } \Gamma_o(\pi) \neq \Gamma(\pi)\}.$$

If V_{0i} is empty, then STOP;

$$P(F_{0i}) = \text{eff}_i(P);$$

otherwise let

$$u_{0i} = \varnothing$$

and

$$u_{1i} = V_{0i} \cup u_{0i}.$$

Suppose now that I_{r-1} and u_r have been defined. Then, the r^{th} iteration is defined as follows.

Iteration I_r

The set S_r consisting of all subpaths of all the elements of the set u_r, is defined as follows.

Definition 10: The set S_r of forbidden paths is defined by the equation:

$S_r = \{ \pi: \pi \in P(G)$ such that π is a subpath of π', $\pi' \in u_r\}$.

This set defines the partition and the costing for iteration I_r. The path-costing function for the r^{th} iteration is now defined by:

Definition 11: $\Gamma_r: P(G) \rightarrow R^+ \times R^+ \times \ldots \times R^+$,where,

$\Gamma_r(\pi) = \Gamma_0(\pi)$ if $\pi \notin S_r$, $\Gamma(\pi)$ otherwise.

Let

$S_{ri} \equiv P_i(G) \cap S_r$, and

$S_{ri}^k = P_i^k(G) \cap S_r$.

Letting

$P(F_{ri}) \equiv \{ \pi: \pi \in P^i(G)$ and $\Gamma_r(\pi) \in (F_{ri})\}$,

the set V_{ri} is defined to be the set of paths whose cost under the costing Γ_r differs from that under Γ.

Definition 12: $V_{ri} = \{\pi: \pi \in P(F_{ri})$ and $\Gamma_r(\pi) \leq \Gamma(\pi)$ and $\Gamma_r(\pi) \neq \Gamma(\pi)\}$.

Backward Dynamic Programming is applied to the set of paths that are in the complement of the set S_{ri}. $F_{ri}^1 = VMIN\{\Gamma_r(\pi): \pi \in P_i^1(G), and\ P \notin S_{ri}\}$. Paths of length at least two in the complement of S_{ri} can take two forms: those all of whose links but the latest come from the previous vector-minimization, and those all whose links except the latest come from a set S_{rj}. Accordingly, vector-minimization over sets of paths of cardinality greater than one must involve, not only paths from the preceding vector-minimization step, but also paths from appropriate forbidden path sets.

$$F_{ri}^{(2)} = VMIN\{\{\Gamma_r(\pi): \Gamma_r(\pi) = f_{rj}^{(1)} + \vec{c}_0(i,j)\ ,\ f_{rj}^{(1)} \in F_{rj}^{(1)}\ ,\ \pi \notin S_{ri}\} \cup$$

$$\{\Gamma_r(\pi):\ \Gamma_r(\pi) = \Gamma_0(\pi') + \vec{c}_0(i,j)\,,\ \pi' \in S_{rj}^{1} \text{ and } \pi \notin S_{ri}\}.$$

The backward Dynamic Programming ends with the evaluation of the set $F_{ri}^{(n-1)}$.

$$F_{ri}^{(n-1)} = \text{VMIN}\{\{\Gamma_r(\pi): \Gamma_r(\pi) = f_{rj}^{(n-2)} + \vec{c}_0(i,j)\,,\ f_{rj}^{(n-2)} \in$$

$$F_{rj}^{(n-2)}\,,\ \pi \notin S_{ri}\} \cup \{\Gamma_r(\pi):\ \Gamma_r(\pi) = \Gamma_0(\pi') + \vec{c}_0(i,j)\,,$$

$$\pi' \in S_{rj}^{n-2} \text{ and } \pi \notin S_{ri}\}\}.$$

Let

$$F_{ri} = \text{VMIN}\,\{F_{ri}^{(n-1)} \cup \{s_{ri}\}\},$$

where

$\{s_{ri}\} \equiv \{\Gamma_r(\pi): \pi \in S_{ri}\}$. Now check the stopping criteria.

If V_{ri} is empty, STOP;

Let $P(F_{ri}) = \text{eff}_i(P)$; otherwise let

$u_{(r+1)i} \equiv V_{ri} \cup u_{ri}$,

$u_{r+1} \equiv \bigcup_{i \neq d} u_{(r+1)i}$.

Theoretical Results

Notation 3: In the sequel, the notation "$\leq \neq$" will stand for "does not exceed, vector-wise, but is distinct from".

Theorem 1: The algorithm terminates after a finite number of iterations.

Proof: With each iteration, the sets u_r strictly increase. This, and the fact that the number of paths in P(G) is finite, yields the desired result.

Let

$P(G_r^i) \equiv P(G) \backslash S_{r'}$

Definition 13: Define $\text{eff}\left(P_i(G_r^i)\right)$ by the condition $\pi \in \text{eff}\left(P_i(G_r^i)\right)$ if and only if the set

$$\left\{\pi': \pi' \in P_i(G_r^i) \text{ and } \Gamma_0(\pi') \leq \neq \Gamma_0(\pi)\right\}$$

is empty. Then we have the following result.

Theorem 2: $P(F_n^{n-1}) = eff(P_i(G_r^i))$.

Proof: (by induction on k, the cardinalty of the path).

Let k = 1. Then,

$$F_n^{(1)} = VMIN\{\Gamma_r(\pi): \pi \in P_i^1(G) \text{ and } \pi \in S_n\};$$

hence one obtains,

$$P(F_{ri}^{(1)}) = eff^{(1)}(P(G_r^i)).$$

Assume that,

$$P(F_{ri}^{(k-1)}) = eff^{(k-1)}(P_i(G_r^i)),\ 1 < k-1 < n.$$

A. Let $\pi \in eff^{(k)}(P_j(G_r^i))$, where π is of cardinality k.

Case 1. $\pi = \{(i,j)\} \cup \pi_j$, $\pi_j \in S_{rj}$. Then it follows that

$\Gamma_r(\pi) = \vec{c}_o(i,j) + \Gamma_o(\pi_j)$,

an element in the set $\{\vec{c}_o(i,j) + \Gamma_o(\pi_j),\ \pi_j \in S_{rj}^{k-1}\}$. Since $\pi \in eff^{(k)}(P_i(G_r^i))$, Since $\pi \in eff^{(k)}(P_i(G_r^i))$, and has cardinality k, $\pi \in P(F_{ri}^{(k)})$, as desired.

Case II. $\pi = \{(i,j)\} \cup \pi_j$, $\pi_j \notin S_r$. We claim that $\pi_j \in eff^{(k-1)}(P_j(G_r^i))$. If not, then there must exist $\pi'_j \in P_j^{(k-1)}(G_r^i)$ such that $\Gamma_r(\pi'_j) \neq \Gamma_r(\pi_j)$. But then, letting $\pi' = \{(i,j)\} \cup \pi'_j$, we get, $\Gamma_r(\pi') \leq\neq \Gamma_r(\pi)$, by the additivity of Γ_r. This is impossible since $\pi \in eff(P_i^{(k)}(G_r^i))$. Hence, $\pi_j \in eff(P_i^{(k-1)}(G_r^i))$. By hypothesis, it follows that $\pi_j \in P(F_{ri}^{(k-1)})$. Therefore, $\Gamma_r(\pi) \in \{\vec{c}_o(i,j) + F_{rj}^{(k-1)}\}$. Since $\pi \in eff(P_i^{(k)}(G_r^i))$, $\Gamma_r(\pi) \in F_{ri}^{(k-1)}$, or $\pi \in P(F_{ri}^{(k-1)})$.

B. Suppose now that $\pi \in P(F_{ri}^{(n-1)}) \backslash eff(P_i^{(k)}(G_r^i))$.

This implies the existence of a path $\pi' \in P(G_r^i)$ such that $\Gamma_r(\pi') \leq\neq \Gamma_r(\pi)$. Without loss of generality, π' can be assumed to be an element of

$\text{eff}(P_i(G'_r))$. But then, by part A, $\pi' \in P(F_{ri}^{(n-1)})$; since $\pi \in P(F_{ri}^{(n-1)})$ by hypothesis, $\Gamma_r(\pi') \leq\neq \Gamma_r(\pi)$ is impossible, by the definition of $F_{ri}^{(n-1)}$.

The theorem follows from parts A and B.

Definition 14: Define the set $\text{eff}_i(P_r)$ by the condition $\pi \in \text{eff}_i(P_r)$ if and only if the set $\{\pi': \pi' \in P_i(G) \text{ and } \Gamma_r(\pi') \leq\neq \Gamma_r(\pi)\}$ is empty.

Theorem 3: $P(F_{ri}) = \text{eff}_i(P_r)$.

Proof:

A. Let $\pi \in \text{eff}(P_i(G_r))$. Note that, $\text{eff}(P_i(G_r)) \subseteq \text{eff}(P_i(G'_r))$.

Case I. Suppose $\pi \in P(G'_r)$. Then, since $\pi \in \text{eff}(P_i(G_r))$, it is also true that, $\pi \in \text{eff}(P_i(G'_r))$; this yields, by Theorem 3, that $\pi \in P(F_{ri}^{(n-1)})$. Since π is an element of $\text{eff}(P_i(G_r))$, and $F_{ri} = \text{VMIN}\{F_{ri}^{(n-1)} \cup \{s_r\}\}$, we must then have $\pi \in P(F_{ri})$.

Case II. Suppose $\pi \notin P_i(G'_r)$. Then, $\pi \in S_{ri}$, by definition, and so must be in $P(F_{ri})$, since it's in $\text{eff}(P_i(G_r))$.

B. Suppose now that $\pi \in P(F_{ri}) \backslash \text{eff}(P_i(G_r))$. Then there must exist $\pi' \in P_i(G_r)$, (WLOG $\pi \in \text{eff}(P_i(G_r)$, such that $\Gamma_r(\pi') \leq\neq \Gamma_r(\pi)$. We now identify two cases:

Case I. $\pi' \in S_{ri}$. Then, $\Gamma_r(\pi') \in \{s_{ri}\}$; but since $\pi \in P(F_{ri})$, we have, by the definition of F_{ri} that $\Gamma_r(\pi') \leq\neq \Gamma_r(\pi)$ is not possible.

Case II. Suppose $\pi' \notin S_{ri}$. Then, $\pi' \in P_i(G'_r)$. Since $\pi' \in \text{eff}(P_i(G'_r))$. we have, by Theorem 3 that $\Gamma_r(\pi') \in F_{ri}^{(n-1)}$. But then $\pi \in P(F_{ri})$ makes $\Gamma_r(\pi') \leq\neq \Gamma_r(\pi)$ impossible.

Parts A and B together establish the theorem.

In the next theorem, the main theoretical result of this paper is established; namely that after a finite number of iterations, the algorithm determines the entire set of efficient paths, for all possible initial nodes.

Theorem 4: Let the algorithm terminate after t iterations. Then,

$P(F_{ti}) = \text{eff}_i(P)$.

Proof: Let $\pi \in \text{eff}_i(P)$. Let $\pi_{(t)}$ denote the path π costed by Γ_t.

Case I. Suppose $\pi_{(t)} \in u_{ti}$. Note that this implies that $\Gamma_t(\pi_{(t)}) = \Gamma(\pi)$. We distinguish two subcases.

Subcase 1. $\pi_{(t)} \in P(F_{ti})$. Here, there is nothing to prove.

Subcase 2. Suppose $\pi_{(t)} \notin P(F_{ti})$.

Since by Theorem 4 it follows that $P(F_{ti}) = \text{eff}_i(P_t)$, this implies that $\pi_{(t)} \notin \text{eff}_i(P_t)$. But then there must exist a path π', WLOG in $\text{eff}_i(P_t)$, such that $\Gamma_t(\pi') \leq\neq \Gamma(\pi)$. Since $\pi' \in P(F_{ti})$, by Theorem 4, and F_{ti} is terminal, $\Gamma_t(\pi') = \Gamma(\pi')$. We thus have, $\Gamma(\pi') = \Gamma_t(\pi') \leq\neq \Gamma_t(\pi) = \Gamma(\pi)$. This is not possible, since $\pi \in \text{eff}_i(P)$.

Case II. Suppose $\pi_{(t)} \notin u_{ti}$. Again, two subcases arise.

Subcase 1. $\pi_{(t)} \in P(F_{ti})$. Here, there is nothing to prove since t is terminal, and so $\Gamma_t(\pi_{(t)}) = \Gamma(\pi)$.

Subcase 2. Suppose $\pi_{(t)} \notin P(F_{ti})$. This implies that there exists π', WLOG in $\text{eff}_i(P_t)$, which by Theorem 4 is the same as $P(F_{ti})$, such that $\Gamma_t(\pi') \leq\neq \Gamma_t(\pi_{(t)})$. But since $\Gamma_t(\pi_t)) \leq \Gamma(\pi_{(t)}) = \Gamma(\pi)$, ($\pi_{(t)}$ and π are identical), we have $\Gamma_t(\pi') \leq\neq \Gamma(\pi)$. However, the facts that $\Gamma_t(\pi') = \Gamma(\pi')$, (recall that $\pi' \in P(F_i)$, t terminal), and $\Gamma_t(\pi') \leq\neq \Gamma(\pi)$ from above, force $\pi \notin \text{eff}_i(P)$. This contradicts the leading assumption.

Now let $\pi \in P(F_{ti}) \backslash \text{eff}_i(P)$. This implies that there exists a path $\pi' \in \text{eff}_i(P)$ such that $\Gamma(\pi') \leq\neq \Gamma(\pi)$. This in turn yields $\Gamma_t(\pi') \leq \Gamma(\pi') \leq\neq \Gamma(\pi)$ But since $\pi \in P(F_{ti})$, $\Gamma_t(\pi) = \Gamma(\pi)$; moreover, by Theorem 4, $\pi \in \text{eff}_i(P_t)$. These two facts make the inequalities $\Gamma_t(\pi') \leq \Gamma(\pi') \leq\neq \Gamma(\pi)$ $(= \Gamma_t(\pi))$ impossible.

The theorem is established.

A Principle of Optimality

The Principle of Optimality of Classical Dynamic Programming is no longer directly applicable to the problem of multi-stage decision optimization with time-dependency, since separability and monotonicity may be violated. There is a generalization of this principle that pertains to this latter problem, a principle that in fact specializes to the classical version in the case of time-invariant parameters. The statement of this generalization requires a rigorous definition of a partition of a set of paths.

Definition 16: Let (P , Γ) denote a set of paths P with costs determined according to the path-costing function Γ. Let Γ_1 and Γ_2 be two path-costing functions defined on two subsets P_1 and $P_2 = P \backslash P_1$ of P. Then, the set $\{(P_1 , \Gamma_1) , (P_2 , \Gamma_2)\}$ is called a partition of (P , Γ).

Definition 17: Let (P' , Γ') be a set of paths costed according to Γ'. A path $\pi \in P'$ is said to be P'-nondominated iff it is nondominated with respect to the set of paths in P' costed according to Γ'.

Theorem 5 (Principle of Optimality): Let (P , Γ) be a given set of paths, as in the algorithm. Suppose $\pi \in P$ is P-nondominated. Then there exists a unique, finite set of partitions, $\{P_r\} = \{\{P_{1r} , \Gamma_o) , (P_{2r} , \Gamma)\}\}$, $P_{1k} \supseteq_{\neq} P_{1(k+1)}$ and $P_{2k} \subseteq_{\neq} P_{2(k+1)}$, and a nonempty set of non-negative integers I_p such that for at least one $i \in I_p$ every subpath of π is P_{1i}-nondominated.

Proof: The algorithm, whose proof appears above, also serves as a constructive proof of this result. An explicit proof follows.

1. Uniqueness: This is established by noting that the initial partition is uniquely defined, consisting of all paths costed at their infima, and that subsequent partitions are uniquely determined by the sets F_{ri}, which are themselves unique.

2. Existence: Let π be P-nondominated. Let s be the smallest index for which $\pi \in P(F_{si}^{(n-1)})$. Such an index exists by the algorithm and finiteness. Suppose now that π' is a subpath of π. If π' coincided with π, there's nothing to prove, since π is, by virtue of being in $P(F_{si}^{(n-1)})$, P_{ls}-nondominated. So suppose $\pi' \neq \pi$. Assume now that there exists a path $\pi'' \in P_{ls}$ such that $\Gamma_o(\pi'') \leq \Gamma_o(\pi')$ and $\Gamma_o(\pi'') \neq \Gamma_o(\pi')$ is true. This yields,

$$\Gamma_o(\{\pi \backslash \pi'\} \cup \{\pi''\}) \leq \Gamma_o(\{\pi \backslash \pi'\} \cup \{\pi'\}) = \Gamma_o(\pi),$$

and

$$\Gamma_o(\{\pi \backslash \pi'\} \cup \{\pi''\}) \neq \Gamma_o(\{\pi \backslash \pi'\} \cup \{\pi'\}) = \Gamma_o(\pi),$$

which is a contradiction.

Note that this is a generalization of the classical Principle of Optimality since in the case of time invariance the set of partitions has cardinality one, and hence every subpath of a nondominated path must be nondominated.

4. A NUMERICAL EXAMPLE

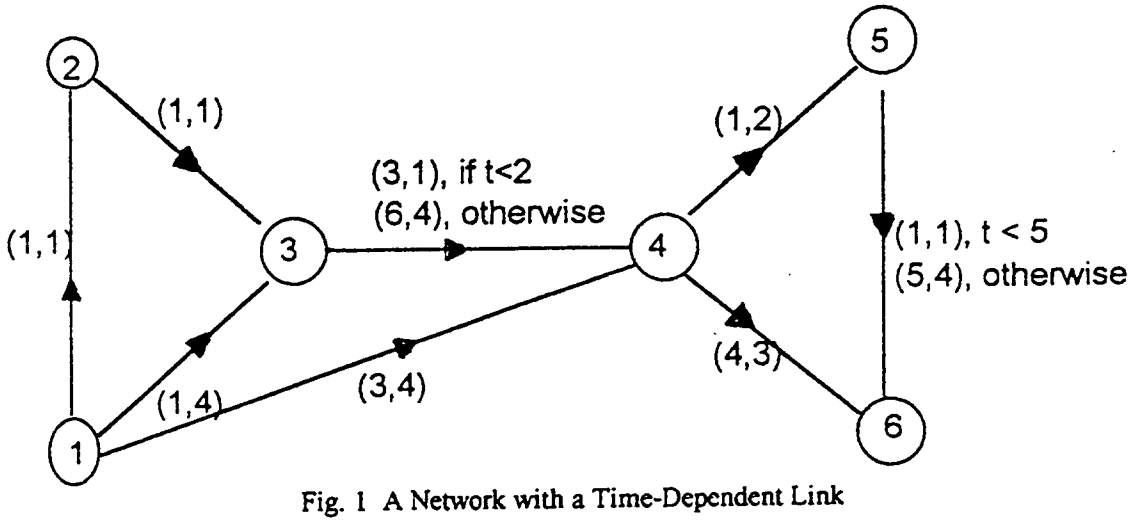


Fig. 1 A Network with a Time-Dependent Link

Initial Iteration

Maximum Path-Length	Start Node	Path	Cost
1	1	------	------
	2	------	------
	3	------	------
	4	4-6	(4,3)
	5	5-6	(1,1)
2	1	1-4-6	(3,4)+(4,3)=(7,7)
	2	------	------
	3	3-4-6	(3,1)+(4,3)=(7,4)
	4	4-6	(4,3)
		4-5-6	(1,2)+(1,1)=(2,3)
	5	5-6	(1,1)
3	1	1-4-6	(7,7)
		1-3-4-6	(1,4)+(7,4)=(8,8)
	2	2-3-4-6	(1,1)+(7,4)=(8,5)
	3	3-4-6	(7,4)
		3-4-5-6	(3,1)+(2,3)=(5,4)
	4	4-5-6	(2,3)
	5	5-6	(1,1)
4	1	1-4-6	(7,7)
		1-3-4-5-6	(1,4)+(5,4)=(6,8)
		1-2-3-4-6	(1,1)+(8,5)=(9,6)
	2	2-3-4-6	(8,5)
		2-3-4-5-6	(1,1)+(5,4)=(6,5)
	3	3-4-5-6	(5,4)
	4	4-5-6	(2,3)
	5	5-6	(1,1)
5	1	1-4-6	(7,7)
		1-3-4-5-6	(6,8)
		1-2-3-4-6	(9,6)
		1-2-3-4-5-6	(1,1)+(6,5)=(7,6)
	2	2-3-4-5-6	(6,5)
	3	3-4-5-6	(5,4)
	4	4-5-6	(2,3)
	5	5-6	(1,1)

Efficient Path	Infimum Cost	True Cost
1-3-4-5-6	(6,8)	(10,11)
1-2-3-4-5-6	(7,6)	(14,12)
2-3-4-5-6	(6,5)	(10,8)
3-4-5-6	(5,4)	(5,4)
4-5-6	(2,3)	(2,3)
5-6	(1,1)	(1,1)

Since the infimum costs and the true costs of the efficient paths from nodes 3,4 and 5 coincide, all the efficient paths from these nodes to the destination node have been determined. Since however this is not the case for the efficient paths from 1 and 2, the avoidance set of paths will be defined. This consists of these paths and all their subpaths.

Initial avoidance set = {1-2-3-4-5-6,2-3-4-5-6,3-4-5-6,4-5-6,5-6,1-3-4-5-6}

Iteration 2: (Backward Dynamic Programming over the complement of the avoidance set)

Maximum Path-Length	Start Node	Path	Cost
1	1	-----	-----
	2	-----	-----
	3	-----	-----
	4	4-6	(4,3)
2	1	1-4-6	(3,4)+(4,3)=(7,7)
	2	-----	-----
	3	3-4-6	(3,1)+(4,3)=(7,4)
	4	4-6	(4,3)
3	1	1-4-6	(7,7)
		1-3-4-6	(1,4)+(7,4)=(8,8)
	2	2-3-4-6	(1,1)+(7,4)=(8,5)
	3	3-4-6	(7,4)
	4	4-6	(4,3)
4	1	1-4-6	(7,7)
		1-2-3-4-6	(1,1)+(8,5)=(9,6)
	2	2-3-4-6	(8,5)
	3	3-4-6	(7,4)
	4	4-6	(4,3)
5	1	1-4-6	(7,7)
		1-2-3-4-6	(9,6)
	2	2-3-4-6	(8,5)
	3	3-4-6	(7,4)
	4	4-6	(4,3)

Note that the efficient paths from nodes 3 and 4 are of no interest, since the entire set has been determined in the initial iteration. We now take the vector-minimum of the union of the set of efficient paths from nodes 1 and 2 and the avoidance set.

VMIN{(7,7),(9,6),(10,11),(14,12)}= {(7,7),(9,6)}; Efficient set = {1-4-6,1-2-3-4-6}
VMIN{(8,5),(10,8)}= (8,5}; Efficient set = {2-3-4-6}

Efficient Path	Infimum Cost	True Cost
1-4-6	(7,7)	(7,7)
1-2-3-4-6	(9,6)	(12,9)
2-3-4-6	(8,5)	(8,5)

The new avoidance path is:

{1-2-3-4-5-6,2-3-4-5-6,3-4-5-6,4-5-6,5-6,1-3-4-5-6,1-2-3-4-6,2-3-4-6,3-4-6,4-6}

Iteration 3:

Maximum Path-Length	Start Node	Path	Cost
1	1	-----	-----
	2	-----	-----
	3	-----	-----
2	1	1-4-6	(3,4)+(4,3)=(7,7)
	2	-----	-----
3	1	1-4-6	(7,7)
		1-3-4-6	(1,4)+(7,4)=(8,8)

4	1	1-4-6	(7,7)
5	1	1-4-6	(7,7)

We now take the vector minimum of the union of the efficient set of paths with the set of avoidance paths from node 1. This yields:
VMIN {(7,7),(12,9),(14,12),(10,11)} = {(7,7)}; Efficient Path = 1-4-6

Efficient Path	Infimum Cost	True Cost
1-4-6	(7,7)	(7,7)

So the algorithm halts.

The entire set of efficient paths is: {1-4-6,2-3-4-6,3-4-5-6,4-5-6,5-6}
It should be noted that even though the path 4-5-6 is an efficient path in its own right, it would have prevented the path 1-4-6 from being manifest (since it dominates the subpath 4-6 of 1-4-6) had it not been avoided in the backward Dynamic Programming in iterations two and three.

5. FUTURE RESEARCH

This algorithm has been shown to be useful in three applications: tactically delayed scheduling, fire egress analysis and production control (see Getachew [12]). One direction of further investigation is the extension of the algorithm to the modeling of tactical delay in the multiple machine case. In considering this extension, one will be faced with be faced with issues of efficient implementation. The single machine case was implemented in SMALLTALK [9], an object-oriented language. Given the requirement in both the first and second phases of the algorithm, for sets to be searched for given subpaths, it was natural that the data structures of an object-oriented language (such as array, bag and dictionary), with exactly this functionality built into them should make such a language a natural choice. It is clear, however, that with increasing problem size, this convenience carries a price. In this connection, it is also of interest to extend the network modeling of tactical delay so that the optima generated are global.

The algorithm, with its alternating Dynamic Programming and avoidance-set-definition phases has been conceived and formulated as an iterative, region-limiting type of algorithm. (It is worth noting that unlike most methods of this type,

it is guaranteed to converge to the optimal solution in a finite number of iterations.) Recent work by Getachew and Kostreva [13] suggests that a very rich application area will be opened by means of this new algorithm, due to its generality and power.

Finally, the algorithm consists of two phases: a backward Dynamic Programming phase, and a forward vector-minimization phase. As it proceeds towards the solution, the time-invariant network of phase one decreases in size while the avoidance set of phase two increases. as such the burden of computation shifts from the combinatorial operations of Dynamic Programming to the lookup and comparison operations of vector minimization. It is of interest to investigate how, given efficient data structures (for the information handling required in both phases), the complexity of this algorithm (for optimization with time variant parameters) compares with Dynamic Programming in the time-invariant case. The conjecture is that, given sufficiently efficient data structures, the two should be comparable to within a polynomial.

Acknowledgments

This research was supported in part by Grant 60NANB0D1023 from the Building and Fire Research Laboratory, National Institute of Standards and Technology, Gaithersburg, Maryland.

References

1. Bellman, R. E., "Dynamic Programming," Princeton University Press, Princeton N.J., 1957.

2. Bellman, R.E., "On a Routing Problem," Quarterly of Applied Mathematics 16 (1958), 87-90.

3. Bellman, R. E. and L. A Zadeh, "Decision-Making in a Fuzzy Environment," Management Science, Vol. 17, No. 4, December, 1970, B-141-B-164.

4. Brown, T. A. and R. E. Strauch, "Dynamic Programming in Multiplicative Lattices," Journal of Mathematical Analysis and Application 12, (1965), 364-370.

5. Brumbaugh-Smith, J. and D. Shier, "An Empirical Investigation of Some Bicriterion Shortest Path Algorithms," European Journal of Operational Research, 43 (1989), 216-224.

6. Cooke, K. L. and E. Halsey, "The Shortest Route through a Network with Time-Dependent Internodal Transit-Times", Journal of Mathematical Analysis and Applications 14 (1966), 493-498.

7. Corley, H. W. and I. D. Moon, "Shortest Paths in Networks with Vector Weights," Journal of Optimization Theory and Applications 46 (1985), 79-86.

8. Daellenbach, H. G. and C. A. DeKluyver, "Note on Multiple Objective Dynamic Programming," Journal of Operational Research Society 31 (1980), 591-594.

9. Digital Inc., "Smalltalk/V Mac Tutorial and Programming Handbook," Digital Inc., Copyright 1988.

10 Dijkstra, E. W., "A Note on Two Problems in Connection with Graphs," Numerische Mathematik 1, (1959), 269-271.

11. Dreyfus, S. E., "An Appraisal of Some Shortest Path Algorithms," Operations Research, 17, (1969), 395-412.

12. Getachew, T., "An Algorithm for Multiple-Objective Network Optimization with Time Variant Link-Costs," Ph.D. Thesis, Management Science, Clemson University, Clemson, South Carolina, 1992.

13. Getachew, T. and M. M. Kostreva, "Transient Behavior in Multiple-Criteria Path-Planning Problems," Proceedings of the 1992 IEEE International Conference on Systems, Man and Cybernetics, Chicago, pages 867-873.

14. Halpern, J., "Shortest Route with Time Dependent Length of Edges and Limited Delay Possibilities in Nodes," Zeitschrift fur Operations Research, Band 21, (1977), Seite 117-124.

15. Henig, M. I., "The Principle of Optimality in Dynamic Programming with Returns in Partially Ordered Sets," Mathematics of Operations Research, 10 (3), (1985), 462-470.

16. Kaufmann, D. E. and R. L. Smith, "Minimum Travel Time Paths in Dynamic Networks with Application to Intelligent Vehicle-Highway Systems, University of Michigan, Transportation Research Institute, IVHS Technical Report-90-11, 1990.

17. Kostreva, M. M. and M. M. Wiecek, "Time Dependency in Multiple Objective Dynamic Programming," Journal of Mathematical Analysis and Applications, 173, (1), (1993), 289-308.

18. Orda, A. and R. Rom, "Shortest-Path and Minimum-Delay Algorithms in Networks with Time-Dependent Edge-Length," Journal of the Association for Computing Machinery 37 (3) (1990), 607-625.

19. Philpott, A.B., "A Class of Continuous-Time Shortest-Path Problems," To appear in SIAM Journal of Control and Optimization.

20. Verdu, S. and H.V. Poor, "Abstract Dynamic Programming Models Under Commutativity Conditions," SIAM Journal of Control and Optimization 25, (4), (1987), 990-1006.

NIST-114
(REV. 6-93)
ADMAN 4.09

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

MANUSCRIPT REVIEW AND APPROVAL

INSTRUCTIONS: ATTACH ORIGINAL OF THIS FORM TO ONE (1) COPY OF MANUSCRIPT AND SEND TO THE SECRETARY, APPROPRIATE EDITORIAL REVIEW BOARD

(ERB USE ONLY)

ERB CONTROL NUMBER	DIVISION
PUBLICATION REPORT NUMBER	**CATEGORY CODE**
NIST-GCR-94-643	
PUBLICATION DATE	**NUMBER PRINTED PAGES**
June 1994	

TITLE AND SUBTITLE (CITE IN FULL)

Mathematical Modeling of Human Egress From Fires in Residential Buildings

CONTRACT OR GRANT NUMBER
60NANB0D1023

TYPE OF REPORT AND/OR PERIOD COVERED
FINAL

AUTHOR(S) (LAST NAME, FIRST INITIAL, SECOND INITIAL)
Michael M. Kostreva
Clemson University
Clemson, SC 29634-1907

PERFORMING ORGANIZATION (CHECK (X) ONE BOX)
[ ] NIST/GAITHERSBURG
[ ] NIST/BOULDER
[ ] JILA/BOULDER

LABORATORY AND DIVISION NAMES (FIRST NIST AUTHOR ONLY)

SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP)
U.S. Department of Commerce
National Institute of Standards and Technology
Gaithersburg, MD 20899

PROPOSED FOR NIST PUBLICATION

[ ] JOURNAL OF RESEARCH (NIST JRES)	[ ] MONOGRAPH (NIST MN)	[ ] LETTER CIRCULAR
[ ] J. PHYS. & CHEM. REF. DATA (JPCRD)	[ ] NATL. STD. REF. DATA SERIES (NIST NSRDS)	[ ] BUILDING SCIENCE SERIES
[ ] HANDBOOK (NIST HB)	[ ] FEDERAL INF. PROCESS. STDS. (NIST FIPS)	[ ] PRODUCT STANDARDS
[ ] SPECIAL PUBLICATION (NIST SP)	[ ] LIST OF PUBLICATIONS (NIST LP)	[X] OTHER NIST-GCR-
[ ] TECHNICAL NOTE (NIST TN)	[ ] NIST INTERAGENCY/INTERNAL REPORT (NISTIR)	

PROPOSED FOR NON-NIST PUBLICATION (CITE FULLY) [ ] U.S. [ ] FOREIGN

PUBLISHING MEDIUM
[X] PAPER [ ] CD-ROM
[ ] DISKETTE (SPECIFY)
[ ] OTHER (SPECIFY)

SUPPLEMENTARY NOTES

ABSTRACT (A 2000-CHARACTER OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, CITE IT HERE. SPELL OUT ACRONYMS ON FIRST REFERENCE.) (CONTINUE ON SEPARATE PAGE, IF NECESSARY.)

Models for predicting human egress in fires have been developed in parallel with those that predict environmental conditions which occur as a fire develops and spreads. Mathematical models which find optimal and/or Pareto optimal (non-dominated) paths for human egress in fires are presented. These optimal paths may serve as multiple functions. In addition to gaining insight into expected and desired actions in fires, such predictions can serve as the basis for directing egress during building evacuation. Optimal paths may serve as a standard against with other heuristically generated paths may be evaluated.

Mathematical models, appropriate solution techniques, coding of the techniques, and sample applications are included.

KEY WORDS (MAXIMUM OF 9; 28 CHARACTERS AND SPACES EACH; SEPARATE WITH SEMICOLONS; ALPHABETIC ORDER; CAPITALIZE ONLY PROPER NAMES)
Computer models; fire models; fire research, hazard assessment; human behavior

AVAILABILITY
[X] UNLIMITED [ ] FOR OFFICIAL DISTRIBUTION - DO NOT RELEASE TO NTIS
[ ] ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GPO, WASHINGTON, DC 20402
[X] ORDER FROM NTIS, SPRINGFIELD, VA 22161

NOTE TO AUTHOR(S): IF YOU DO NOT WISH THIS MANUSCRIPT ANNOUNCED BEFORE PUBLICATION, PLEASE CHECK HERE. [X]

WORDPERFECT